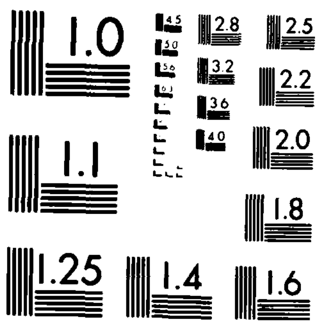
AD-A152 137 SOLUTION TECHNIQUES FOR WHOLESALE PROVISIONING OF REPLACEMENT PARTS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W A GOULDING SEP 84 1/1

UNCLASSIFIED F/G 15/5 NL

END



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

SOLUTION TECHNIQUES FOR
WHOLESALE PROVISIONING
OF REPLACEMENT PARTS

by

William A. Goulding

September 1984

Thesis Advisor: G. T. Howard

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A152137	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Solution Techniques for Wholesale Provisioning of Replacement Parts		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis September 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) William A. Goulding		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		12. REPORT DATE September 1984
		13. NUMBER OF PAGES 84
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Repo-

18. SUPPLEMENTARY NOTES

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Wholesale Provisioning, Dynamic Programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The purpose of this thesis is to present solution techniques for Provisioning problems arising in the Navy's wholesale purchase of replacement parts. The objective is to minimize the Mean Supply Response Time (MSRT) subject to a budget constraint. The problem can be formulated as a Dynamic Program (DP), however, it is too large and complex for a standard recursive dynamic approach. Consequently, a

DD FORM 1473, 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

S N 0102-LF-014-6601

 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (Continued)

variation of the normal DP approach was developed that significantly reduces the required computations. An existing DP computer program was modified to implement this DP variation. The result is a useable approach considering speed and ease of manipulation.

A-1

Approved for public release; distribution unlimited.

Solution Techniques for Wholesale Provisioning of Replacement Parts

by

William A. Goulding
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1973

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
SEPTEMBER 1984

Author: William A. Goulding
William A. Goulding

Approved by: G.T. Howard
G.T. Howard, Thesis Advisor

F.R. Richards
F. R. Richards, Second Reader

A.R. Washburn
A. R. Washburn, Chairman,
Department of Operations Research

K.T. Marshall
K. T. Marshall,
Dean of Information and Policy Sciences

ABSTRACT

The purpose of this thesis is to present solution techniques for provisioning problems arising in the Navy's wholesale purchase of replacement parts. The objective is to minimize the Mean Supply Response Time (MSRT) subject to a budget constraint. The problem can be formulated as a Dynamic Program (DP), however, it is too large and complex for a standard recursive dynamic approach. Consequently, a variation of the normal DP approach was developed that significantly reduces the required computations. An existing DP computer program was modified to implement this DP variation. The result is a usable approach considering speed and ease of manipulation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. HISTORY AND INTRODUCTION

The U. S. Navy is a hardware oriented organization requiring a wide variety of spare and replacement parts. The Naval Supply Systems Command (NAVSUP) is the organization within the Navy responsible for maintaining a sufficient stock of parts and replenishing this stock when quantities run low. NAVSUP is keenly interested in the quality of the models it uses to provide support for the Navy's weapons systems. Faculty of the Naval Postgraduate School were asked to examine the existing wholesale and retail provisioning and replenishment models and to provide comments on their correctness and suggestions for possible improvements. In response, a report by Richards and McMasters [Ref. 1] addresses the problems and various solution techniques. This thesis examines a specific solution technique that may be applicable to one of the problems. The following is a brief synopsis of the existing models and the environment in which they operate.

Within NAVSUP there are two centers that use provisioning models, 1) Ships Parts Control Center (SPCC) and 2) Aviation Supply Office (ASO). Presently the Inventory Control Points (ICPs) within the centers have two retail provisioning models which generate, 1) COSALS (Coordinated Shipboard Allowance Lists) and 2) AVCALS (Aviation Coordinated Allowance Lists). These models supply sufficient parts support for a short term ship deployment and try to provide protection from replacement part depletion until the wholesale replenishment models can buy or repair the needed spare parts.

Weapon system procurement requires that provisioning support be available by a set preliminary operational

capability (POC) date. The initial support requirements are determined by a wholesale provisioning model. The support must be comprehensive enough to supply a new weapon system until the replenishment buy can be made and delivered. It should be noted that the provisioning lead time is usually two to two and one half years. Variability in production and installation causes the quality of the forecasted demand to be poor. Hence, the provisioning buy may result in large quantities of excess parts in the supply system, or there may be many stockouts before the replenishment buy becomes available. The Assistant Secretary of Defense (Installations and Logistics) has provided guidance [Ref. 2] to help prevent large excesses. From this instruction a third model (COSDIF) was developed to conservatively estimate the quantities of new parts required for wholesale provisioning. For existing equipment no additional stock is procured for the new weapons system. If the increase in demand from the new system is large enough, the inventory manager can anticipate the increase and start a replenishment buy in sufficient time to meet the need. Otherwise, inventory optimization models may be used to control the stock. Wholesale replenishment for a new system is initiated at different times by the two centers within NAVSUP. For either center the reorder point for replacement parts is subject to change as a function of the actual demand the weapon system places on the supply system. The reorder point for an existing system occurs at part depletion levels set by NAVSUP.

Budgeting constraints for support of existing systems are partially driven by the peculiarities of the Department of Defense Planning, Programming and Budgeting System (PPBS). Those purchases for new weapon systems are funded as part of the procurement process. For existing systems, each Service requests from Congress sufficient funding to

meet specific levels of readiness and availability set by the respective Service Head. NAVSUP receives this direction from the Chief of Naval Operations (CNO) and then prioritizes the parts required with a procedure called "Variable Threshold". This technique considers several inputs peculiar to the weapon system to develop the priority list for items to be purchased. The quantity (depth) to be purchased is determimed from historical data or from engineering estimates of the number of anticipated failures for a period of time equal to one procurement lead time plus one quarter. Proceeding from the top of the priority list the parts are purchased to their 'depth' until the budget is expended. Those parts unfunded for purchase are placed in a file for Unfunded Requirements.

Obviously, the optimal quantities to purchase, as a function of the inputs, and the maximum utilization of the budget are of high interest. Howard in [Ref. 3] addresses several different approaches that provide solutions for this type problem. The thrust of this thesis is a variation of one of those solution techniques for a similarly defined provisioning problem. The objective is to minimize the Mean Supply Response Time (MSRT) subject to a budget constraint.

The overall MSRT is a nonlinear composite function of the purchase quantity for all items. The individual item's MSRT functions are also nonlinear with performance as a function of four variables peculiar to the item type. The budget constraint is a linear function with constant costs for each item and a set budget limit. There are no cost advantages for large magnitude buys or item combination buys. The individual costs and return functions of the MSRT lend themselves to formulation as a dynamic programming (DP) problem. However, the problem is large and complex for a standard recursive dynamic approach due to the required number of stages, the magnitude of the budget and the range

of costs. Consequently, a variation of the normal DP approach was developed that significantly reduces the required computations. This variation shall be referred to as 'the funnel' because of its funneling characteristics for computation of the resource variable bounds. Further explanation of its construction and procedure may be found in Chap 2&3. Modifications of an existing DP computer program described in [Ref. 3] to operate as the DP variation makes the 'funnel' method a usable approach considering speed and ease of manipulation. Additionally, the DP variation has the advantage of providing an optimal integer solution. In this report 'the funnel' will be compared to a full DP procedure and a computerized marginal analysis approach which operates very efficiently but which may not generate the optimal solution.

II. SOLUTION TECHNIQUES

This chapter will discuss three methods for solving the provisioning problem described in Chapter 1, 1) marginal analysis, 2) pure dynamic programming, and 3) a modified dynamic programming procedure. The first technique provides a quick solution that is a good estimate but may not be optimal. The second technique provides a global optimal solution but requires a large amount of computer time for any moderate sized problem. The third technique provides a local, and possibly a global, optimal solution and requires significantly less time than option two.

The following definitions will be used in the formulation and discussions in this chapter. Capital letters represent the variable while lower case letters indicate a subscript. In the expression Xi-1 the quantity i-1 is the subscript although it appears on the same line as the symbol X.

N	= the total number of items considered for provisioning
B	= pre-assigned maximum budget amount
Ci	= the unit cost of item i
Ei	= the essentiality code for item type i
Ki	= the fixed time required to satisfy a requisition for item i if stock is available
Li	= the demand rate for item i (lambda)
Si	= the decision variable for the number of items of type i
Ti	= the procurement lead time for item i
P'i(Si)	= the probability of Si demands for item i during the provisioning interval

$Pi(Si)$ = the probability of Si or fewer demands for item i during the provisioning interval

$Zi(Si)$ = the performance measure for item i when Si units are stocked.

The objective of this problem is to minimize the Mean Supply Response Time (MSRT) as a function of Si. MSRT is a widely used measure of supply performance because of its role in determining availability and because it is an indicator of the success of a supply system in meeting response time goals. It considers two major factors in computation,

1) the likelihood of satisfying demands from stock on hand and
2) the length of the delay in satisfying demands when the system runs out of stock.

It may be represented by the following expression,

$$MSRT=\min \sum Ei*Li*Ti*Zi(Si)/\sum Ei*Li*Ti \qquad (2.1)$$

$$i=1,\ldots,N,$$

where the likelihood adjustments for conditions one and two are factors in the Zi(Si) function. This function represents the performance measure of item i when Si items are considered and is defined by the following equation for the case where demands during the provisioning leadtime are assumed to be Poisson distributed:

$$Zi(Si)=Ki+((1-Pi(Si))*(Li*Ti-2Si+Si*(Si+1) \qquad (2.2)$$

$$/Li*Ti)+P'i(Si)(Li*Ti-Si))/2Li).$$

The constraint for this problem is the budget or the resource in dollars. It is linear and may be represented by the following inequality,

$$\sum Ci*Si<=B \quad i=1,...,N. \tag{2.3}$$

The following subsections present solution techniques which are viable procedures for solving the above formulated provisioning problem.

A. MARGINAL ANALYSIS

The marginal analysis method is a quick, simple and effective approach for generating a solution to the previously defined provisioning problem. For the problem considered here, the marginal analysis approach generates solutions which may not be optimal. However, the solutions generated by the procedure do possess some very desirable properties which are stated below.

Let Zi(Si) be the essentiality weighted MSRT for item i when Si units are stocked. Let ΔZi(Si)=Zi(Si-1)-Zi(Si) be the reduction in essentiality weighted MSRT achieved by increasing the stockage level for item i from Si-1 units to Si units. Let S'=(S1,S2,...,SN) be the vector of allocations for the N items: i.e. Si is the number of units allocated to item i. The marginal approach begins with a budget of B dollars and an allocation vector S'=(0,0,...,0). It computes ΔZi(1)/C1 for i=1,2,...,N and selects that item for which this marginal benefit to cost ratio is greatest. Without loss of generality, assume that the item selected is item 1. This requires an expenditure of C1 dollars and results in the allocation vector S'=(1,0,0,...,0). The procedure next computes ΔZ1(2)/C1, the marginal benefit to cost ratio achieved by increasing the stockage for item 1 by one unit from a level of one to two units. The maximum value is then selected from among the ratios in 2.4.
An additional unit of the item, say item j, for which the maximum occurs is selected and the remaining budget is

$$\frac{\Delta Z1(2)}{C1}, \frac{\Delta Z2(1)}{C2}, \frac{\Delta Z3(1)}{C3}, \ldots, \frac{\Delta ZN(1)}{CN} \qquad (2.4)$$

decremented by Cj dollars. The process continues in this manner allocating a single unit iteratively until the budget is expended or no additional units can be purchased. At each step, the ratios compared are

$$\frac{\Delta Z1(S1+1)}{C1}, \frac{\Delta Z2(S2+1)}{C2}, \ldots, \frac{\Delta ZN(SN+1)}{CN}, \qquad (2.5)$$

where the current allocation vector is S'=(S1,S2,...SN). Since only one of the ratios changes at each step of the process, the computations required are very few.

It can be shown [Ref. 4], that for the problem considered here the solution is undominated. That is, if B(S') is the budget required to fund the allocation vector S', and if S'' is any other allocation vector such that B(S'')<=B(S') then the overall essentiality weighted MSRT for S' is smaller than that for S''. If it should happen that B(S')= B, the original budget, for some step of the marginal analysis procedure, then the solution S' is optimal.

In actual practice, what usually happens in the marginal analysis procedure is that at some step purchase of the item with the largest marginal benefit to cost ratio requires an expenditure of more money than that which remains. The allocation vector at that point is not feasible and various heuristics are usually applied to continue the process. The most popular heuristic is to backtrack to the previous feasible solution and select that item which has the maximum benefit to cost ratio among those items with unit costs no larger than the budget remaining. On applying such heuristics, one can not claim that the final solution so obtained is undominated. However, one can make use of the undominated property to obtain a bound on the optimal value of the objective function.

Let Z(SS) be the objective function value at the occasion at which the marginal analysis procedure first generates a solution for which the budget required is equal to or greater than B. If equality exists, SS is optimal. Suppose B(SS)=BS>B and let S*(X) be the optimal solution for a budget of X dollars. Then, since SS is undominated, Z(SS)<=Z(S*(BS))<=Z(S*(B)), which is the value of the objective function at the optimal solution. Thus Z(SS) is a lower bound on the optimal value of the objective function. One can thus assess the potential benefit of a more extensive search for the optimal solution by comparing the actual performance obtained at any stage with the lower bound discussed above.

A computer program was written by Richards to implement this method. A sample of this Marginal Analysis program output in table I shows the iteration number, which items are purchased, the numbers of items purchased, the benefit to cost ratios and the budget remaining. The table shows the sequence in which the items are purchased. Note the decreasing benefit to cost ratio. The program selects item number one five times in a row before the first unit of item two provides a better marginal return. The last attempt to purchase chooses an item that drives the remaining budget negative. The program automatically re-evaluates the marginal returns and then chooses the next largest benefit to cost ratio which in this case drives the remaining budget to zero. As previously discussed this solution may not be optimal.

The Marginal Analysis program has several other capabilities based on the same approach. There is a job selection menu that provides a choice of 1) spares determination, 2) budget determination or 3) kit evaluation. The first option selects the quantities to be purchased within a given budget. The second option solves a related problem by

TABLE I

Sample Output from the Marginal Analysis Program

```
**************************************************
*     METHOD: FIXED      BUDGET =       205.00    *
**************************************************

**************************************************
   STEP  ITEM  SI       RATIO                BR
     1     1     1   0.480808496          200.00
     2     1     2   0.365659833          195.00
     3     1     3   0.260618091          190.00
     4     1     4   0.172421157          185.00
     5     1     5   0.105280340          180.00
     6     2     1   0.0735758543         170.00
     7     3     1   0.0600002967         155.00
     8     1     6   0.0591956675         150.00
     9     3     2   0.0533369593         135.00
    10     3     3   0.0466887541         120.00
    11     3     4   0.0400909968         105.00
    12     3     5   0.0336193480          90.00
    13     1     7   0.0306577347          85.00
    14     3     6   0.0273999237          70.00
    15     3     7   0.0216008648          55.00
    16     2     2   0.0207276568          45.00
    17     3     8   0.0164023377          30.00
    18     1     8   0.0146531016          25.00
    19     3     9   0.0119544677          10.00
    20     3    10   0.0083406679          -5.00
    20     1     9   0.0064818673           5.00
    21     1    10   0.0026624922           0.0

**************************************************
MINIMIZE MSRT   EXCESS:   0.0

    I        SI      MSRT
    1        10      0.190
    2         2     10.396
    3         9     12.176

         OVERALL   5.385

 REALIZED PERFORMANCE =      5.3852
 BOUNDS (    6.0033,     4.3120)
```

computing the required budget to meet a specified constraint, in this case MSRT. The third option evaluates the available performance for a given decision set. The program will conduct the optimization problem with a LaGrange Multiplier or marginal analysis. In either case the answer is the same. The speed and ease of manipulation of this procedure are desirable; however, to be guaranteed an optimal solution a means of verification must be found. That is the goal of this thesis.

B. STANDARD DYNAMIC PROGRAMMING

The standard dynamic programming approach can be applied to this type of provisioning problem. The following sample problem illustrates how the concept may be applied. A single stage problem corresponding to a single item may be characterized as in fig 2.1.

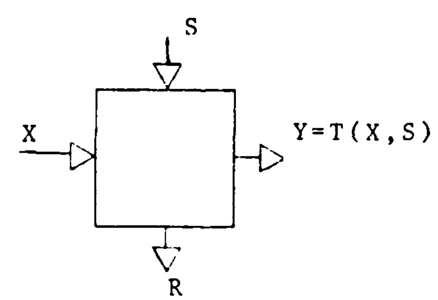


Figure 2.1 Example of a One-Stage Decision Problem.

In figure 2.1,

X is the input state variable which is the resource available for use at this stage.

S is the decision variable that represents the decision

within the stage.

R is the stage return function that produces a return based on the input X and the decision S.

Y is the output state which is a function of X and S.

T is the stage transformation function that expresses the components of the output state Y as a function of the input state and decisions, $Y=T(X,S)$.

An '*' after an Xi or Si variable indicates the optimal value for that stage, [Ref. 5 p22-23.].

The one-stage initial-state optimization problem is to find the best stage return as a function of the input state X. The optimal return from the stage will be denoted as F(X) and the optimal decision as S*(X). Thus,

$$F(X)=\min_{S(X)} R(X,S(X))=R(X,S^*(X)), \qquad (2.6)$$

where the decision S*(X) provides the best return for that particular input value X. If X were to change the values of S*(X) and the return function should also change.

Now consider a multistage problem as shown in figure 2.2. This is the standard recursive DP structure with backward numbering where stage N is on the left and the stage 1 on the right. The optimal return for stages i,i-1,...,1 is represented by Fi(Xi). The solution technique solves succeeding stages from right to left providing the optimal return FN(XN) up to the current stage under consideration. In the single stage problem Y was the output state variable, but now Y serves as the input to the next stage so Xi-1=Y=Ti(Xi,Si). Thus, in an N stage problem, the output from stage i becomes the input to stage i-1 for i=1,...,N. In figure 2.2

Xi is the state variable

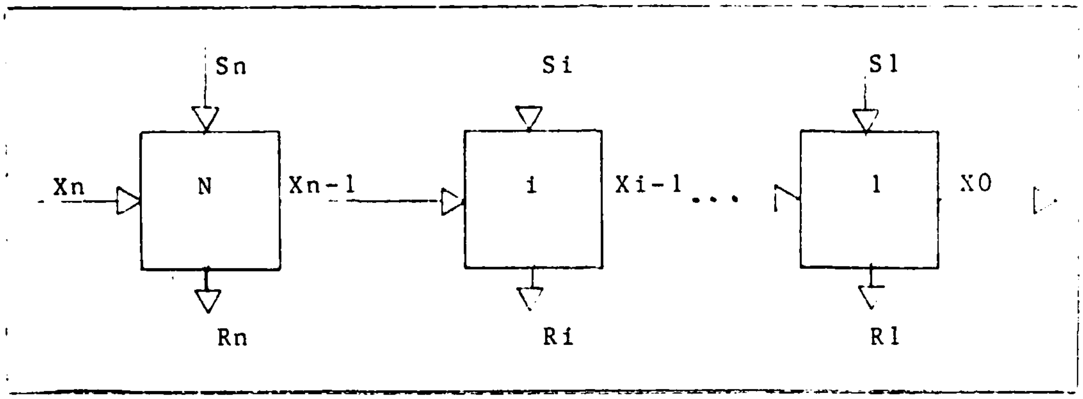


Figure 2.2 Example of an N Stage Decision Problem.

Si is the decision variable
Ri is the stage return function
Ti is the stage transformation function
* after an Xi or Si variable indicates the optimal value for that stage,

where i=1,...,N is the stage index. For each stage a decision Si*(Xi) will be made that yields the optimal return Fi(Xi) as a function of the present stage return R(Xi,Si*(Xi)) and the optimal return Fi-1(Xi-1) for the stages 1 through i-1. The recursive equation for a minimization procedure would be,

$$Fi(Xi)=\min_{Si} (Ri(Xi,Si)+Fi-1(Xi-1)) \quad i=1,\ldots,N, \tag{2.7}$$

where

$$Xi-1=Ti(Xi,Si)=Xi-CiSi \quad i=1,\ldots,N. \tag{2.8}$$

The stage transformation is simply the budget minus the quantity purchased multiplied by its cost. This recursion assumes the overall return function is the weighted sum of

the individual return functions for all replacement parts. The equation for this relationship is,

$$MSRT=\sum Ri(Xi,Si) \quad i=1,...,N. \qquad (2.9)$$

The problem is to make a series of decisions SN...S1 that will optimize the overall return function while remaining within the budget constraints.

For the previously discussed provisioning problem, using the different replacement parts as the stages, the state variable Xi is the resource, i.e., the dollars remaining at each stage. The decision variable Si is the number of parts purchased in stage i. The stage return function is the MSRT which must be computed from four operating characteristics for each part, (essentiality, procurement lead time, demand rate, performance as a function of quantity procured). The following equation represents the return function at each stage,

$$Ri(Xi,Si)=Ei*Li*Ti*Zi(Si)/\sum Ei*Li*Ti \qquad (2.10)$$

$$i=1,...,N.$$

The Standard DP approach computes the function Fi(Xi) from equation 2.7 recursively for i=1,...,N and for all possible values of Xi. At each stage i this is done by examining all feasible decisions for each Xi, computing the return for each decision, and then adding the corresponding optimal return Fi(Xi-1) from the remaining stages to arrive at the optimal return for the present Xi. Then for all Xi values in the stage, the best total return is chosen and assigned to Fi(Xi). At the Nth stage the process is complete and FN(XN) represents the optimal return for the problem. This process is efficient for small problems, however it becomes computationally more difficult as the

number of stages and the number of possible state variable values increase in each stage.

For an example of the computational burden, consider a very simple problem with a budget B, a constant increment K for the state variable Xi within each stage, and a max of D decisions for each state variable value. In this case the number of evaluations for one stage is equal to (B/K)*D. Assuming the transformation function and the recursive equation for the optimal return equation require only one addition each, then there are 2*(B/K)*D additions per stage. Each state variable value will require D-1 comparisons, hence, there are (B/K)*(D-1) comparisons per stage. The total number of operations (additions plus comparisons) required for all stages may be represented by,

$$N((2*B*D)/K)+(B*(D-1)/K), \tag{2.11}$$

or simplified this becomes

$$(N*B/K)*(3D-1). \tag{2.12}$$

This example is actually a gross over-simplification of the computational problem. It is possible the return function could be very complex requiring many operations for each decision at each Xi value through the range of the budget. However, for a numerical comparison, consider a 25-stage problem with a budget of 74000, a state variable increment of 1000 and a maximum of 100 decisions. This leads to 553150 operations as shown in equation 2.13,

$$((25*74000)/1000)*((3*100)-1)=553,150. \tag{2.13}$$

This is the required number of additions and comparisons for a 25 stage problem. For today's high speed computer an

addition operation at machine language level takes about twenty-four machine cycles while a comparison takes eighteen. Using twenty cycles as an average and a machine speed of two million cycles per second, the above number of comparisons and additions would take approximately nine seconds. This time estimate is for an incredibly simple problem that does not take into account any of the administrative overhead, objective or constraint computations or increased calculations required by complex functions. By setting K=10, which is equivalent to reducing the stage increment from 1000 to 10, equation 2.14 shows that the number of operations will increase to

$$((25*74000)/10)*((3*100)-1)=55,315,000. \qquad (2.14)$$

Hence, it is easy to see the limitation of standard dynamic programming. While it gives integer solutions, the computational burden is considerable even in simple problems.

C. A DYNAMIC PROGRAMMING VARIATION

The computational workload of the full dynamic programming procedure makes it undesirable for large problems. A technique that will be referred to as 'FUNNELING' because of its control of the resource boundaries between the stages, was developed and applied to a standard recursive DP computer program. The result is a shortened procedure that provides a locally optimal integer solution to problems that can be formulated in the DP construct.

For any optimal solution to an N-stage DP problem there is a series of decisions Si* for i=1,...,N. Corresponding to this solution there is a sequence of Xi* values for i=1,...,N. These optimal state variable values Xi*, represent the best quantities of resource available for

consumption at that stage and are not known in advance. If it were possible to find an approximate value for Xi* in each stage, the optimal solution then could be localized with an iterative process. That is, rather than compute F(X) for all X values there could be, a 'window' of interest, centered around the approximate Xi* value in each stage. The computations would be performed on the limited range of Xi values in that window thereby reducing the required workload significantly. This concept of controlling computational boundaries around an approximate optimal Xi* value is the foundation for the DP variation.

The problem initially is how to localize the regions of the best Xi. This is solved by the first run of the funnel program using a large state variable increment. The resulting Xi values are then used as an approximation to the initial Xi* values i=1,...,N. A subroutine of the funnel program then constructs a window around the area of interest for further refinement. The first estimate of the Xi* values is quite rough and the decisions, especially the Si which have costs less than the initializing increment, may change significantly as the Xi are examined with a lesser state variable increment.

The second program run is the first refinement of the initial solution. Using the large increments the program will have chosen the best return available though it may not be close to the optimal value. The increased sensitivity of the reduced state variable increment will allow consideration of more Xi values in the window permitting the optimization process to move closer to the best return. It is conceivable that any Xi may change as much as the window of its stage will allow. This much variation in the decisions indicates that the best solution lies outside the window and the boundaries must be changed accordingly. This situation may be present in several stages for any program iteration.

Due to the characteristics of the DP procedure, a change in the state variable consumed at stage N will cause the quantity of resource available at stage N-1 and succeeding stages to change. The concept is that repetitive runs of the funneling program will allow the computational bounds for each stage to localize and center upon their respective X_i^*. When each of the stages has chosen its X_i^*, the S_i^* for $i=1,\ldots,N$ will represent the best solution for a given objective and budget. This solution will be a local and possibly a global optimum. Further discussion of the program specifics may be found in Chap 3.

Additional computations may be saved in the funneling program by ranking the stages, largest to smallest, by the cost of their respective parts. Purchasing the most expensive items first will reduce the magnitude of the budget and simplify the computations in the remaining stages. The amount of computation depends on three things, 1) the width of the window for each stage, 2) the number of state variable increments within the window and 3) the number of decisions available for each state variable increment.

Consider again the computational example of section B, where the number of operations is estimated for the dynamic programming solution procedure. To use that analysis for the funnel procedure, B in equation 2.12 is replaced by the width of the window W. This width will vary between stages but this example will assume W_i is constant and equal to 10000. Setting the remaining factors the same, K=10, S=100, and N=25, the number of operations required from equation 2.12 is,

$$((25*10000)/10)*((3*100)-1)=7,475,000. \qquad (2.15)$$

This is roughly 13.5% of the full DP requirement computed in equation 2.11, a marked improvement in efficiency. There

are other ways to make the algorithm more efficient and save computations. Further discussion of the details of the modified DP procedure may be found in chapter 3.

III. THE FUNNEL PROGRAM

This chapter discusses the operation of the standard DP program, the modifications required to employ the funneling concept and some of the implementation problems encountered in the process. Specific subroutine operations are explained in the second section.

A. MODIFICATION FOR THE FUNNEL PROCEDURE

The DP variation, called the Funnel program, is a modified version of a standard DP computer program named DP5. The modified program will be referred to as DP6 for the sake of clarity in the following discussion. The DP5 program as described in [Ref. 3] has four subroutines, STGRET, STORE, TRANFM and DLIMIT. The following list describes what function each subroutine performs in the DP process.

1) STORE - enters the constants to be used in the other subroutines and the main program.

2) DLIMIT - defines the range of the decision values Si for any value of Xi in stage i.

3) STGRET - defines and computes the return function for any values of Xi and Si in stage i, Ri(Xi,Si).

4) TRANFM - defines and computes the transformation function for any values of Xi and Si in stage i, Ti(Xi,Si).

Providing DP5 with the specific input data as required in [Ref. 3], the program will compute the optimal return by the standard recursive DP procedure. The following figure shows the flow diagram for the general solution technique.

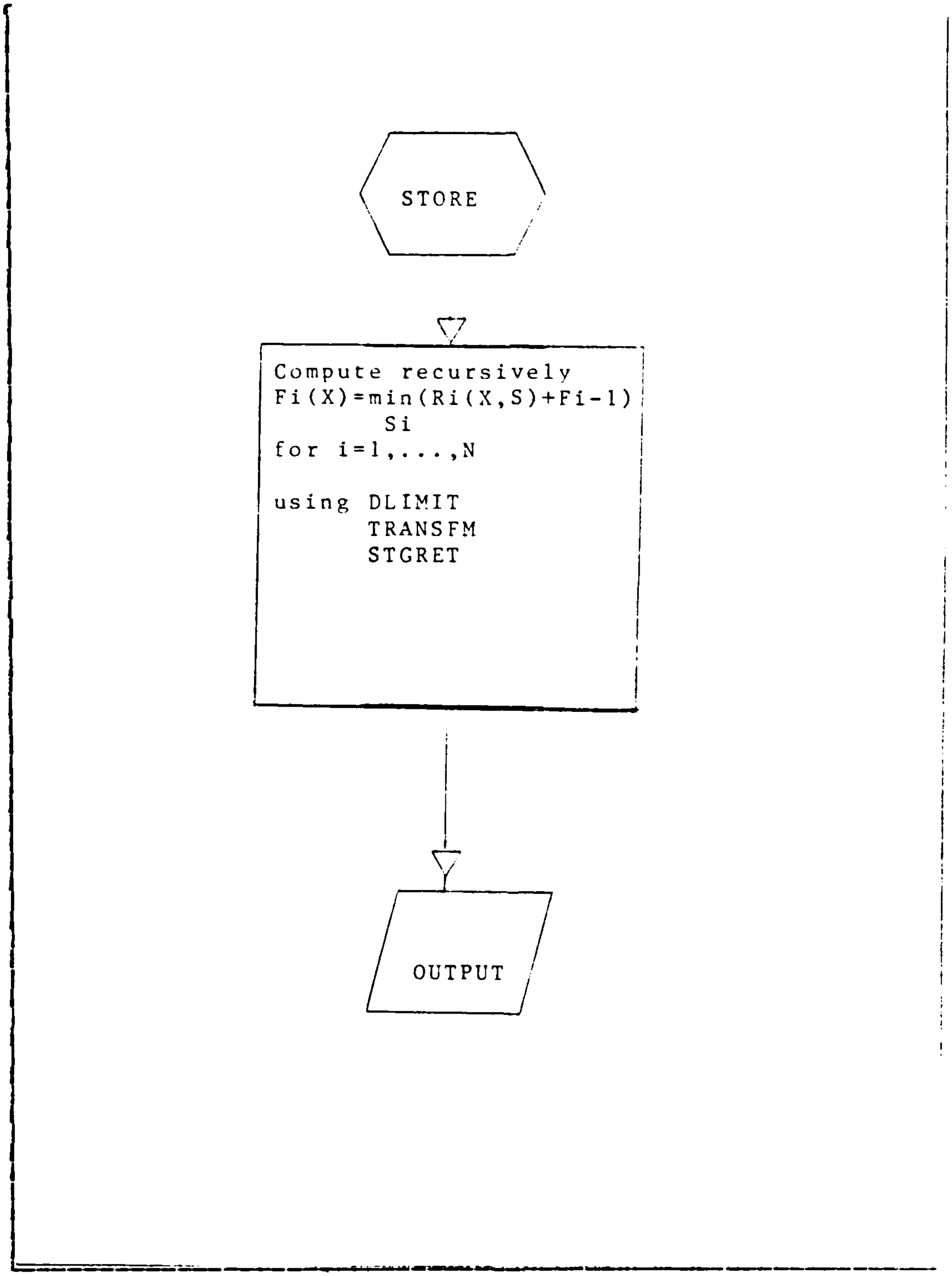


Figure 3.1 Flow Diagram of the DP5 Procedure.

The input data are the parameters for the particular problem being solved. When DP5 is run it uses the STORE subroutine to place the parameters in memory, then calls the other subroutines, DLIMIT, TRANFM, and STGRET to perform the DP procedure.

To use the concept of funneling described in Chap 2, additional subroutines must be added to DP5 to control the state variable computational boundaries in each of the stages. There are two possible starting conditions for the funnel procedure, 1) some input information is available for the decisions at each stage, or 2) no prior knowledge of the decisions is available. To respond to these conditions two subroutines ENTER and ADJUST have been created to correctly manipulate the inputs. The following list describes what function each of the subroutines performs in the Funnel program.

1) ENTER - reads a decision set from a file then creates the input data file in the correct format for the Funnel program,

2) ADJUST - manipulates the computational boundaries of an input data file to reflect the changes required at each program iteration.

The general procedure for DP6 is shown in figure 3.2, The input parameters are read into memory with the STORE function. Then, depending on the starting condition, the ENTER function may used to create an input data file for the Funnel program from a previously available solution. This solution may be obtained from, 1) some other solution method, 2) a rule-of-thumb estimate of the solution, or from 3) past information on similar problems. In any of these cases the estimated or best guess decision set is transformed to the correct input format for the Funnel program.

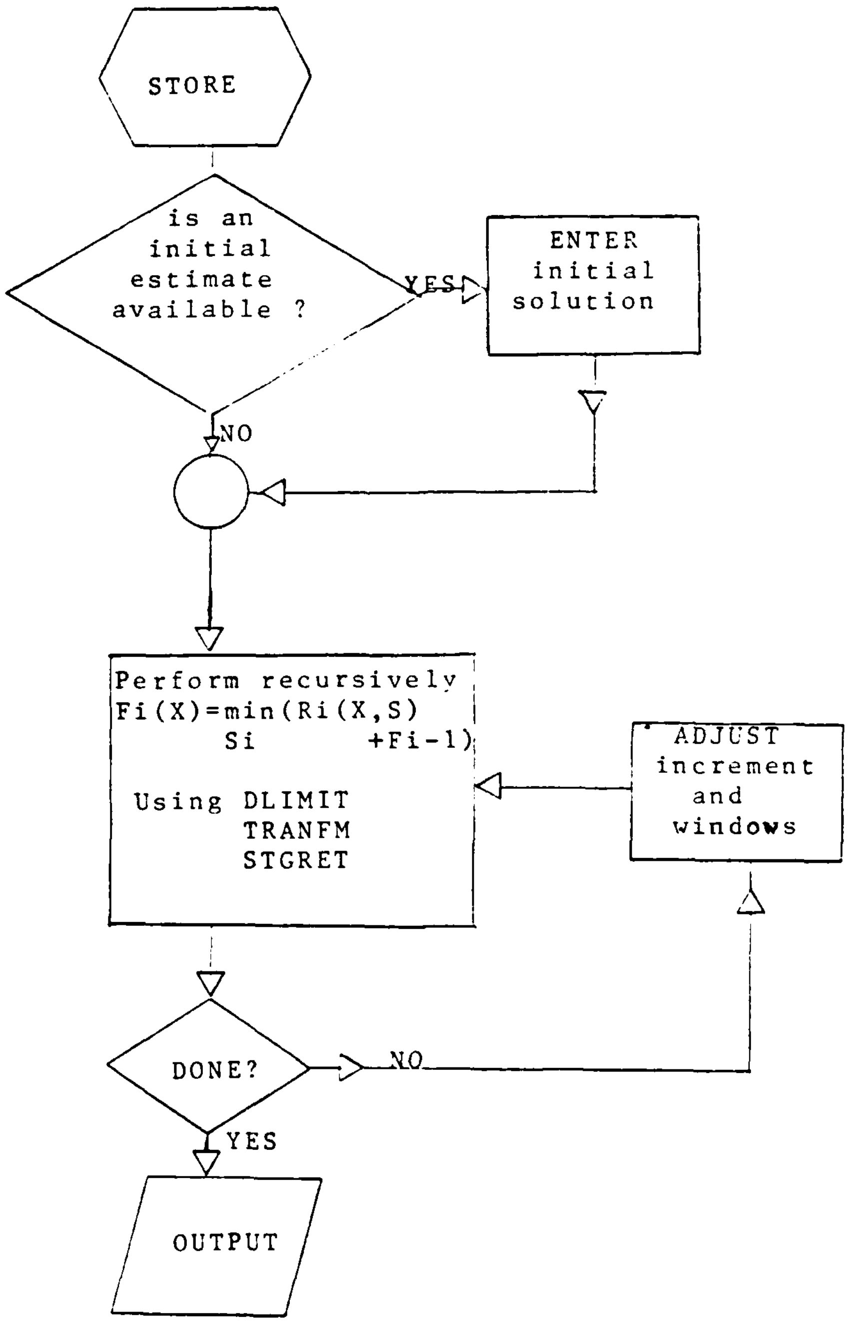


Figure 3.2 Flow diagram of the DP6 procedure.

When there is no previous solution, the beginning data file may be constructed in the correct format as defined by the user instructions located in the comment section of DP6, see appendix. This input file, as described, may be used directly with the program. The next step in the figure is the first iteration of the Funnel process utilizing the DLIMIT, TRANFM, and SIGRET subroutines. When the calculations are complete the results must be examined to see if the process has found the optimal solution, i.e., 'is it DONE'. The requirements for the 'DONE' condition are 1) an all integer solution, 2) no window boundaries violated and 3) the budget remaining must be less than the smallest cost for items to be purchased. If all of these conditions are not met the ADJUST subroutine is activated and the input data file is manipulated to correct the problem areas. With each program iteration this corrective procedure is repeated until all the conditions are met and an optimal solution is produced. An example of the Funnel program output is shown is figure 3.3.

```
THE OBJECTIVE IS MSRT
CONSTRAINT ON COST

THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE    1
APPLIES TO STAGE    1 THRU STAGE    3 INCLUSIVE
THE PROBLEM IS TO MINIMIZE A    3 STAGE PROCESSXN
IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=2.05000D+02
AND XN=2.0500D+02
OPTIMAL XN=2.05000D+02 OPTIMAL RETURN=5.30246D+00
 N         XN                SN              XN-1
 3     2.0500D+02        1.00D+01        5.5000D+01
 2     5.5000D+01        2.00D+00        3.5000D+01
 1     3.5000D+01        7.00D+00        0.0
```

Figure 3.3 Example of DP6 Output.

The objective and constraint are stated in the first two lines. The description that follows gives the number of stages, the budget quantity, the optimal return and detailed stage results. Specifically delineated for each of the stages are the input budget quantities, the decisions and the output budget quantities. For this particular example the budget remaining at the end of the problem is zero.

Two ancillary functions were added, one to the main program and one to a subroutine to provide information to the user. The first is a timing function that provides the program run time. The second is a counting function that presents to the screen and stores in a file the numbers of the stages in which the optimal return values are too close to the computational boundaries. Both functions have provided valuable insights that will be discussed in Chapter 5.

B. THE SUBROUTINE ADJUST

The subroutine ADJUST is the heart of the funneling technique. To gain the computation reduction desired, this subroutine manipulates the windows around the 'current' state variable values, Xi', in each stage for each program iteration. Initially the upper and lower limits HXi' and Lxi' on the window at stage i are defined by,

$$HXi' = Xi'+K*Ci+1, \quad (3.1)$$

and

$$LXi' = Xi'-K*Ci+1, \quad (3.2)$$

where K is a constant. These bounds are illustrated in fig 3.4. Further adjustment of these limits may be necessary to

assure that one of the Xi values considered in the computation of F(Xi) is Xi'. At stage i in the computation of F(Xi) the optimization considers first Xi=LXi, then Xi is incremented by Ci to give the next value, etc. Thus if LXi is not properly set, the value Xi=Xi' will never be considered.

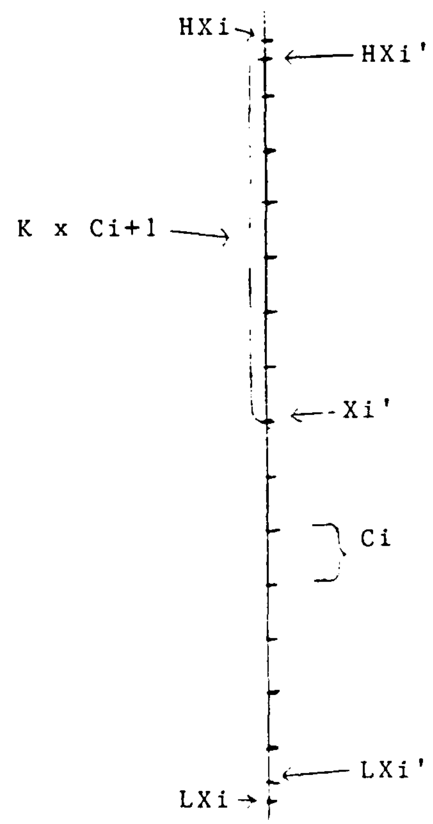


Figure 3.4 Example of a Window on a Stage i.

This adjustment of LXi' is made by decrementing from the temporary stage value, Xi', by the cost Ci for that stage. When this decrementing process first yields a value less than or equal to LXi' that value is set as the new lower limit on the window and is called LXi. A similar adjustment is made to the upper limit by incrementing from Xi' until the value HXi' is equalled or exceeded. The new upper limit is called HXi. This process is repeated for all the stages until the funnel is formed. In figure 3.5, the funnel may be seen for N stages of a provisioning problem. The budget quantity B is represented by the vertical lines for all stages. The upper and lower bounds are represented by HXi, LXi for i=1,...,N where i is the stage index. In each stage the bounds are at least a distance equal to the product of the constant K and Ci+1 from the Xi' value. Stages where the bounds are further from the center indicate the additional adjustment for compatability between the successive Xi' values.

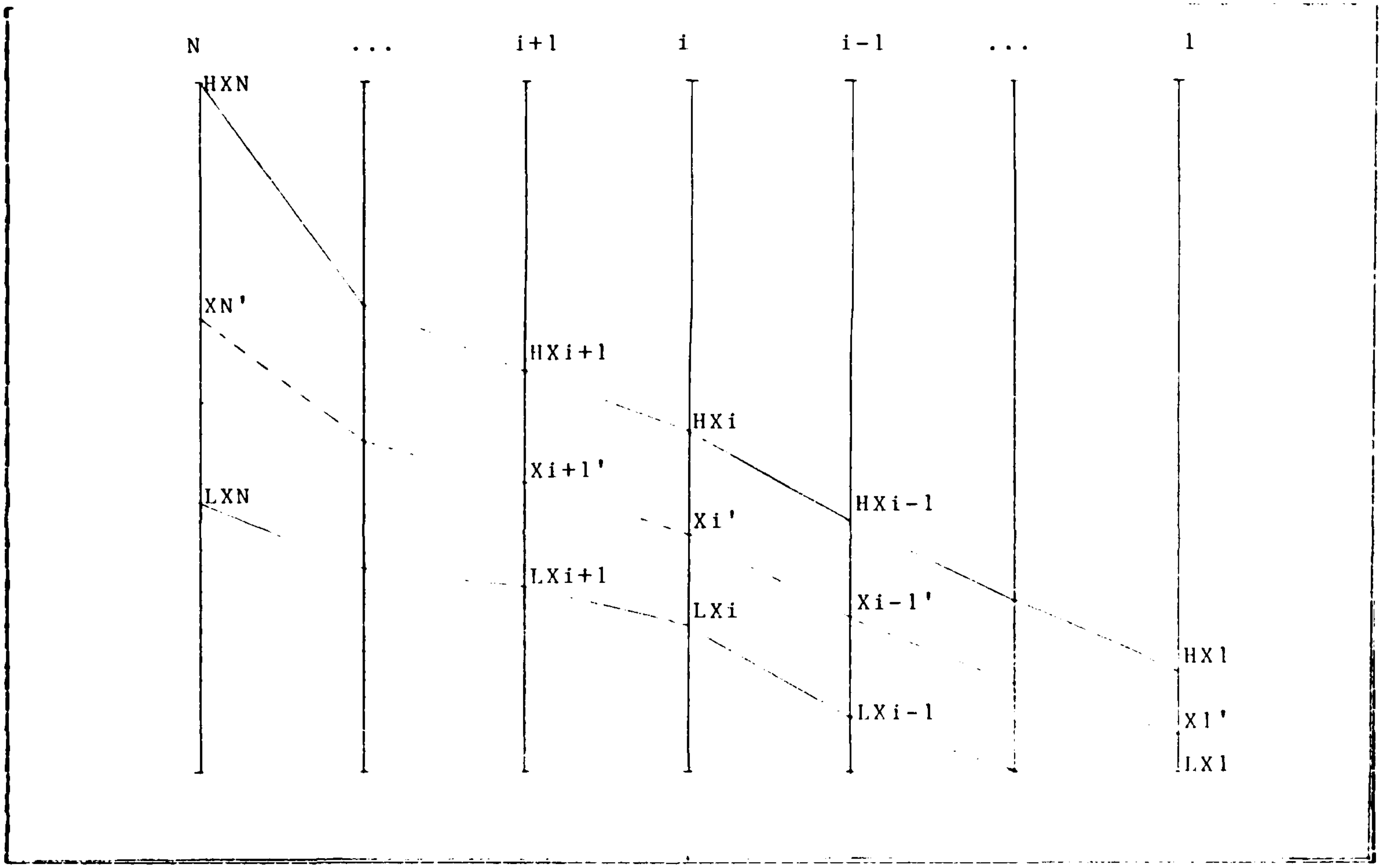


Figure 3.5 A Funnel Diagram for N Stages of a Problem.

The reason for multiplying the succeeding stage costs by the constant K is to allow the DP procedure some computational freedom in each of the stages while searching for the Xi and Si*(Xi) values. During the computational process, should a stage choose an Xi' that is within Ci+1 of the computational bound, the boundary is said to be violated. This is because the DP procedure may actually prefer a value outside the bound but is unable to reach it. Figures 3.6 through figure 3.9 show an example of the stage boundary manipulation as the iterative process of the DP6 program searches for the optimal solution.

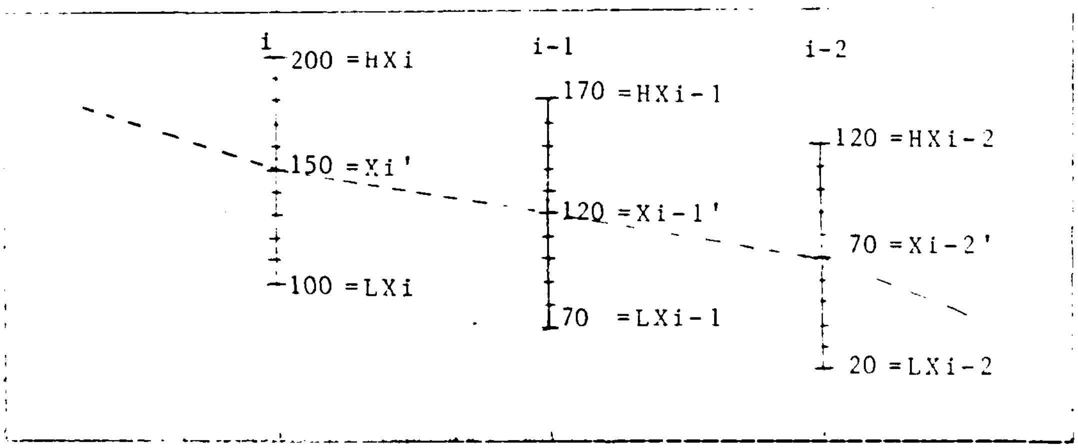


Figure 3.6 The Initialization Run.

The initializing run depends on the starting condition for the problem. If the ENTER function is used, the state variable increments will represent the costs of the respective stages. Figure 3.6 shows a starting condition of no prior knowledge; hence, the state variable increment is a constant, in this example 10, for all stages and of equal number on either side of the current Xi' estimate.

A second run of the program provides an opportunity for the DP process to evaluate the estimated optimal returns with a

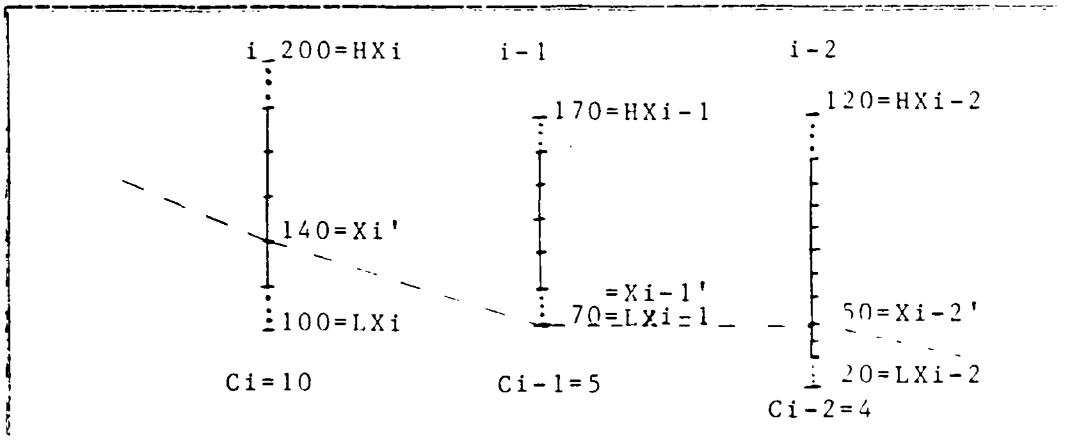


Figure 3.7 The Second Run.

smaller state variable increment or a revised boundary value and to adjust the window around a new Xi' accordingly. Figure 3.7 shows the funnel after the second run but before the windows are adjusted. The program has selected Xi'=140, Xi-1'=70, and Xi-2'=50. These are the current estimates of the optimal Xi' values at each stage. Notice that stage i-1 has a boundary that is violated, i.e., the Xi-1' for the stage is within Ci=10 of the boundary. In this case the optimization function may prefer an Xi-1 outside the window but the boundary prevents its selection. Also in the second run the state variable increments have been changed to the cost of the item associated with the stage. In most stages these values are smaller then the initial increment making the return functions more sensitive. This increase in sensitivity should improve the optimization selection.

The violation of the lower boundary in stage i-1 causes the iteration to fail the conditions of the DONE checkpoint of figure 3.2. Consequently, the ADJUST subroutine is activated again to manipulate the boundaries producing the results shown in figure 3.8.

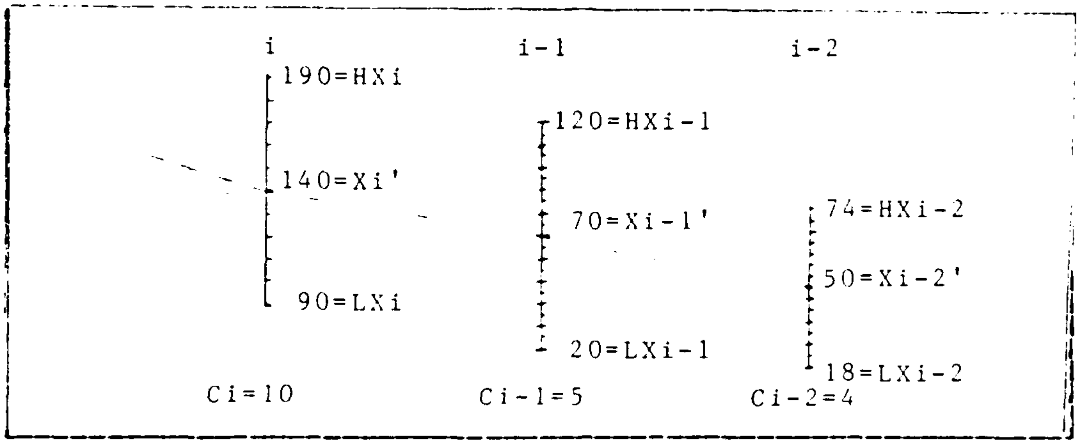


Figure 3.8 The Window Adjustments.

The points in this figure represent the input file for the third run of the program. In stage i-1 the window boundaries are centered on the previous iteration LXi-1 value. Notice the window bounds for stage i-2 have automatically adjusted downward to compensate for the increased resource consumption and avoid cascading boundary violations.

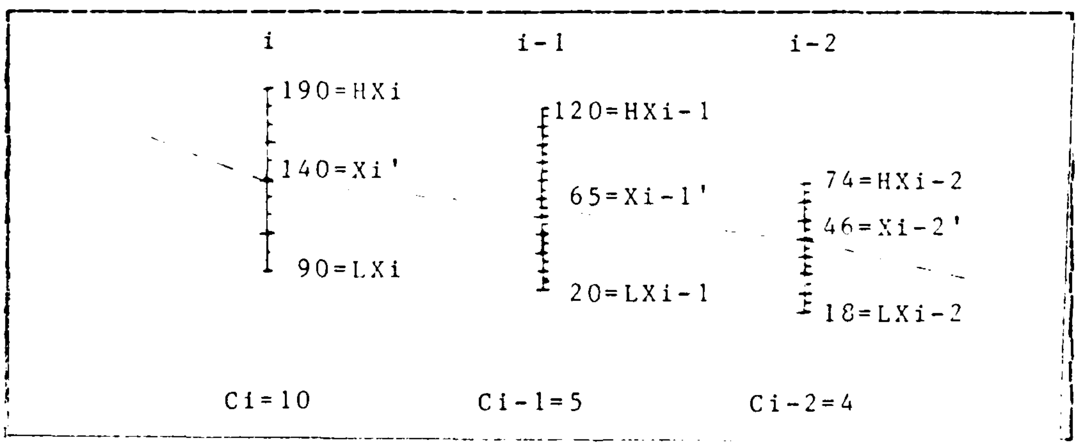


Figure 3.9 The Third Run.

Figure 3.9 shows the funnel after the third run. The program has selected Xi'=140, Xi-1'=65 and Xi-2'=46 for the current estimates of the Xi' values at each stage. Notice that the Xi-1' value is just outside the previous LXi-1, which was 70. In this example the Funnel program manipulated the boundaries of the windows to allow the DP process to choose different Xi' values for the three stages, i, i-1, i-2. Subsequent runs of the program will determine the Xi* for each stage. The process is 'DONE' when the remaining budget <= minimum Ci, an optimal integer solution is provided, and there are no boundaries violated.

C. IMPLEMENTATION PROBLEMS

From the characteristics of the funneling process, there are problems that arise when transitioning between stages. When computing the maximum and minimum value of the decision to be made for each Xi in a stage i, the upper and lower bounds and item cost of stage i-1 must be considered.
In figure 3.10, the left side is the window for stage i, the right side is the window for stage i-1. For the stages i, i-1, the highest values of the state variables are represented by HXi and HXi-1 while the lowest values are represented by LXi, LXi-1. In the program DP6, the two variables DLOW and DHIGH represent the minimum and maximum decisions available for the state variable in each stage. It is essential that DHIGH for any Xi value in the window of the ith stage be of such a magnitude that it will not force the Xi-1 value below the lower bound LXi-1. Additionally, DLOW must be of such magnitude that for any Xi in the window of the ith stage, it will not force the Xi-1 value to violate the upper bound HXi-1. If either of these decision quantities is incorrect at stage i the resource quantity will be outside the window at stage i-1. The derivation of the

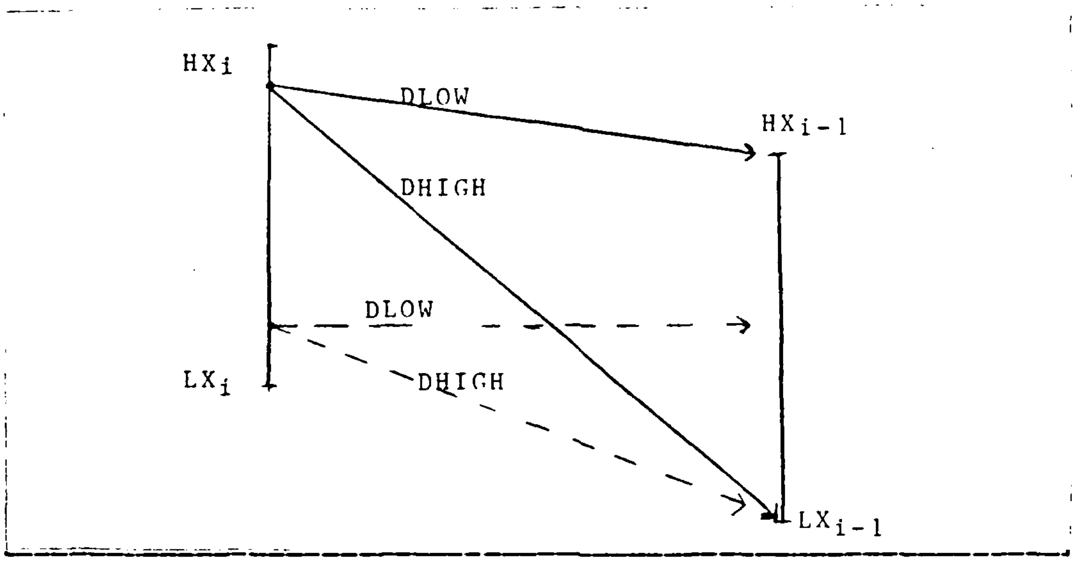


Figure 3.10 Boundary Manipulation.

formulas for these values follows. The transformation function as represented by equation 2.8 of Chap 2 is,

$$X_{i-1}=X_i-C_iS_i \quad i=1,\ldots,N. \tag{3.3}$$

If this transformation is to take place from a window in stage i to a window in stage i-1 the following conditions must be true.

$$LX_{i-1} <= X_i-C_i*SHIGH, \tag{3.4}$$

where SHIGH is the maximum number of items to be purchased and

$$X_i-C_i*SLOW <= HX_{i-1}, \tag{3.5}$$

where SLOW is the minimum number of parts to be purchased. By algebraic manipulation a relationship for SLOW and SHIGH may be established,

$$SHIGH <= (Xi-LXi-1)/Ci, \tag{3.6}$$

and

$$SLOW >= (Xi-HXi-1)/Ci. \tag{3.7}$$

The expressions SHIGH AND SLOW are equivalent to the DHIGH, DLOW required in DP6 and are coded in the subroutine DLIMIT to control the boundary calculations. It is conceivable that at the extreme state variable values of Xi the computed boundaries could become negative or larger than the assigned budget. Thus, the decisons are further limited by DHIGH= min(SHIGH,200) and DLOW= max(0, SLOW), where 200 is an arbitrarily chosen value.

The initialization for the problem is either from input decisions or else a large state variable increment is used. After the subroutine ADJUST manipulates the windows the first time, the state variable increments are decreased or the boundary conditions are changed to allow examination of the new possible Xi values for the best return. The subroutine DLIMIT automatically adjusts the DHIGH and DLOW values to compensate for the changes in the state variable increment. With this closer scrutiny, the optimal return value may change. The ADJUST subroutine moves the boundaries up or down accordingly when an optimal value is within one increment of the bounds. Recall that if the optimal return value is within one increment of window bound, it may be that the DP process would prefer a value outside the window. Hence, the boundary is recorded as violated, and the window is moved for the next iteration. This process is repeated until all the optimal return values are centered within the windows for all stages.

To save computations and avoid computing Xi values that are not compatible between stages, the state variable

increment for each stage is the cost of the item to be purchased in the previous stage. A problem arises in that the original DP5 program may provide solutions that are not integer. This is possible because in the stage being considered, the best return may require an Xi' value from succeeding stages that has not been computed. DP5 handles this by interpolating linearly between the state variable values it does have to arrive at an estimate for the one required. In DP6 to avoid this approximation and obtain integer solutions, the subroutine ADJUST, (as discussed in section A), manipulates the lower and upper window values to ensure the previous stage optimal return values are available for the Xi being considered. This adjustment process is also repetitive and is done in conjunction with the boundary control.

Another problem arises from the effort to save computer computation time. If there is a wide range of costs, the use of the item cost as the state variable increment within a stage becomes troublesome. Initially the ADJUST subroutine creates the state variable windows for the individual stages using a constant multiplier that may be set by the user. To remedy the problem of the extreme range of the costs of items for the provisioning problem being considered, ($10000.00 to .50), the multiplier was changed to a variable that depends directly on the item cost. This was done to decrease the program repetitions required to reach an optimal solution. Under the constant multiplier method, a change in the purchase quantity of a larger cost item, (i.e.$1500), could require many program repetitions to re-allocate the resource available for the lower cost items, (i.e.$10,5,3.75, etc). The smaller window range, due to the constant multiplier, hampers the re-allocation effort. The variable multiplier allows greater compensation for lower cost items to facilitate a more rapid change if required.

IV. ANALYSIS

The primary topic of this chapter will be the comparison of the Marginal Analysis and Funneling solution techniques for the previously stated problem. The discussion will cover the results of both programs and the solution difference. The standard dynamic program procedure will not be addressed because of the computational advantage of the DP variant and the similiarity of the two techniques.

For simplicity and ease of comparison a small three stage problem will be defined and solved using both solution procedures. As in the original provisioning problem the objective is to minimize the MSRT with a constraint on cost. Three different budget quantities will be examined to demonstrate the differences in the solution techniques. The following table is a list of the input parameters which are cost Ci, demand rate Li, essentiality Ei, and procurement lead time Ti. For this particular example Ti is one year.

TABLE II
Sample Problem Input Parameters

I	DEMAND	COST	T	ESSEN
1	5.000	5.00	1.00	3.0
2	1.000	10.00	1.00	2.0
3	10.000	15.00	1.00	1.0

For the first budget quantity of $195.00 the Marginal Analysis program returns the following output.

TABLE III

Marginal Analysis Program Return, Budget=$195.00

```
**************************************************
*    METHOD: FIXED      BUDGET =       195.00    *
**************************************************

**************************************************
   STEP ITEM   SI     RATIO                BR
    1     1    1   0.480808496          190.00
    2     1    2   0.365659833          185.00
    3     1    3   0.260618091          180.00
    4     1    4   0.172421157          175.00
    5     1    5   0.105280340          170.00
    6     2    1   0.0735758543         160.00
    7     3    1   0.0600002967         145.00
    8     1    6   0.0591956675         140.00
    9     3    2   0.0533369593         125.00
   10     3    3   0.0466887541         110.00
   11     3    4   0.0400909968          95.00
   12     3    5   0.0336193480          80.00
   13     1    7   0.0306577347          75.00
   14     3    6   0.0273999237          60.00
   15     3    7   0.0216008648          45.00
   16     2    2   0.0207276568          35.00
   17     3    8   0.0164023377          20.00
   18     1    8   0.0146531016          15.00
   19     3    9   0.0119544677           0.0


**************************************************
MINIMIZE MSRT   EXCESS: 0.0

   I       SI       MSRT
   1        8       1.302
   2        2      10.396
   3        9      12.176

        OVERALL    6.003

 REALIZED PERFORMANCE =      6.0033
 BOUNDS (    6.0033,     6.0033)
```

The decision results are, S1=8, S2=2, S3=9 and the return is Z= 6.0033, with no budget remaining. Viewing the purchase sequence in table III it may be seen that the Marginal

Analysis program has reached a solution uninterrupted, that is without implementing the heuristic developed in [Ref. 1]. Consequently, for this budget quantity the solution is optimal. Notice another indicator of the optimal solution in the last line of table III. The bounds described are the possible upper and lower values for the return. When these quantities are the same value, the solution is the best available.

TABLE IV

Funnel Program Results, Budget=$195.00

```
THE OBJECTIVE IS MSRT
CONSTRAINT ON COST

THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE  1
APPLIES TO STAGE    1 THRU STAGE    3 INCLUSIVE
THE PROBLEM IS TO MINIMIZE A   3 STAGE PROCESSXN
IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=1.9500D+02
AND XN=1.9500D+02
OPTIMAL XN=1.95000D+02 OPTIMAL RETURN=6.00332D+00
   N        XN                 SN              XN-1
   3     1.9500D+02         9.00D+00        6.0000D+01
   2     6.0000D+01         2.00D+00        4.0000D+01
   1     4.0000D+01         8.00D+00        0.0
```

The same budget and parameters entered in the Funnel program produce the results in table IV. The output is in the same format discussed in Chapter 3. The initial budget quantity, XN and the optimal return are shown in the line just above the individual stage results. The variable Xi is the budget quantity at the beginning of the stage, Si is the decision or quantity purchased, and Xi-1 is the budget at the end of the stage. Recall that item 3 is the most expensive and item 1 the least expensive. The optimal solution for the $195.00 budget is S1*=8, S2*=2, and S3*=9 with a Z value equal to 6.0033 and no budget remaining. This answer

matches exactly the Marginal Analysis program results shown in table III, confirming that the Marginal Analysis solution is optimal.

TABLE V

Marginal Analysis Program Return, Budget=$200.00

```
**************************************************
*     METHOD: FIXED       BUDGET =       200.00     *
**************************************************

**************************************************
    STEP  ITEM   SI       RATIO                BR
      1     1     1   0.480808496          195.00
      2     1     2   0.365659833          190.00
      3     1     3   0.260618091          185.00
      4     1     4   0.172421157          180.00
      5     1     5   0.105280340          175.00
      6     2     1   0.0735758543         165.00
      7     3     1   0.0600002967         150.00
      8     1     6   0.0591956675         145.00
      9     3     2   0.0533369593         130.00
     10     3     3   0.0466887541         115.00
     11     3     4   0.0400909968         100.00
     12     3     5   0.0336193480          85.00
     13     1     7   0.0306577347          80.00
     14     3     6   0.0273999237          65.00
     15     3     7   0.0216008648          50.00
     16     2     2   0.0207276568          40.00
     17     3     8   0.0164023377          25.00
     18     1     8   0.0146531016          20.00
     19     3     9   0.0119544677           5.00
     20     3    10   0.0083406679         -10.00
     20     1     9   0.0064818673           0.0

**************************************************
MINIMIZE MSRT   EXCESS:   0.0

    I        SI      MSRT
    1         9      0.514
    2         2     10.396
    3         9     12.176

          OVERALL   5.565

 REALIZED PERFORMANCE =      5.565
 BOUNDS (    6.0033,    4.3120)
```

For the second example the budget quantity of $200.00 was used. The Marginal Analysis program returns the output

in table V. The results are S3=9, S2=2, and S1=9 with a Z value of 5.5652 and no budget left over. Examining the detailed output of the Marginal Analysis program, table V, it may be seen that the purchase of the twentieth item drives the budget negative. The program implements the heuristic, re-evaluating the benefit to cost ratios, then chooses the number one item for the next purchase. Notice that in the bottom line of table V the upper and lower bounds are not equal. This is a good indication the solution may not be optimal and should be verified.

TABLE VI

Funnel Program Results, Budget=$200.00

```
THE OBJECTIVE IS MSRT
CONSTRAINT ON COST

THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE    1
APPLIES TO STAGE     1 THRU STAGE     3 INCLUSIVE
THE PROBLEM IS TO MINIMIZE A   3 STAGE PROCESSXN
IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=2.0000D+02
AND XN=2.0000D+02
OPTIMAL XN=2.0000D+02 OPTIMAL RETURN=5.56519D+00
   N         XN                SN            XN-1
   3      2.0000D+02        9.00D+00       6.5000D+01
   2      6.5000D+01        2.00D+00       4.5000D+01
   1      4.5000D+01        9.00D+00       0.0
```

The same budget and parameters entered in the Funnel program produce the results given in table VI. In this case the answer is again identical, S1=9, S2=2, and S3=9 with the optimal return Z= 5.56519. Incidentally, the heuristic in the Marginal Analysis program makes the correct choice at the twentieth step when it selects the item with the second highest benefit to cost ratio.

For a third case, a budget of $205.00 is entered in both programs. The results for the Marginal Analysis code are shown in table VII.

TABLE VII

Marginal Analysis Program Results, Budget=$205.00

```
*********************************************
*    METHOD: FIXED     BUDGET =     205.00    *
*********************************************

*********************************************
```

STEP	ITEM	SI	RATIO	BR
1	1	1	0.480808496	200.00
2	1	2	0.365659833	195.00
3	1	3	0.260618091	190.00
4	1	4	0.172421157	185.00
5	1	5	0.105280340	180.00
6	2	1	0.0735758543	170.00
7	3	1	0.0600002967	155.00
8	1	6	0.0591956675	150.00
9	3	2	0.0533369593	135.00
10	3	3	0.0466887541	120.00
11	3	4	0.0400909968	105.00
12	3	5	0.0336193480	90.00
13	1	7	0.0306577347	85.00
14	3	6	0.0273999237	70.00
15	3	7	0.0216008648	55.00
16	2	2	0.0207276568	45.00
17	3	8	0.0164023377	30.00
18	1	8	0.0146531016	25.00
19	3	9	0.0119544677	10.00
20	3	10	0.0083406679	-5.00
20	1	9	0.0064818673	5.00
21	1	10	0.0026624922	0.0

```
*********************************************
```

MINIMIZE MSRT EXCESS: 0.0

I	SI	MSRT
1	10	0.190
2	2	10.396
3	9	12.176

OVERALL 5.385

REALIZED PERFORMANCE = 5.3852
BOUNDS (6.0033, 4.3120)

The solution is S1=10, S2=2, and S3=9 with a return value of Z= 5.3852. The upper and lower bounds are not equal indicating the heuristic has been applied and the solution should be verified. Examining the output from the Marginal Analysis program, table VII, it may be seen the budget was again driven negative at the twentieth item. The heuristic

was activated purchasing two of the number one items to drive the budget to zero.

Entering the same input and budget parameters in the Funnel program produces the results shown in table VIII.

TABLE VIII

Funnel Program Results, Budget=$205.00

```
THE OBJECTIVE IS MSRT
CONSTRAINT ON COST

THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE    1
APPLIES TO STAGE     1 THRU STAGE     3 INCLUSIVE
THE PROBLEM IS TO MINIMIZE A    3 STAGE PROCESSXN
IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=2.05000D+02
AND XN=2.0500D+02
OPTIMAL XN=2.05000D+02 OPTIMAL RETURN=5.30246D+00
   N        XN              SN             XN-1
   3     2.0500D+02      1.00D+01       5.5000D+01
   2     5.5000D+01      2.00D+00       3.5000D+01
   1     3.5000D+01      7.00D+00       0.0
```

The answers are not the same. The Funnel program results indicate the best part selection is S1=7, S2=2, and S3=10 for an optimal return of Z= 5.30246. This selection provides an MSRT that is roughly .08 better than the marginal analysis method.

Recall from section A of Chapter 2 that the Marginal Analysis program has a function called KIT that will determine the return from a given decision set. Entering the optimal solution from the Funnel program into the KIT function produces the result in table IX. The resulting objective value is identical to the value computed by the DP variant. This shows that the computational procedures are the same for both programs.

Now consider the definition of a much larger problem. It is of the same type as the previous examples, an

TABLE IX

Kit Function Results, Budget=$205.00

EVALUATION OF SPECIFIED ALLOCATION....

I	SI	MSRT
1	7	3.085
2	2	10.396
3	10	7.610
	OVERALL	5.302

objective of minimizing MSRT with a constraint on cost. The input parameters cost Ci, demand rate Li, essentiality Ei, and procurement lead time Ti, are shown in in table X. The budget is $74825.00 with twenty five stages and costs ranging from $10000.00 to .50. Due to the length of the detailed results an abbreviated output of the Marginal Analysis solution is shown in table XI. The return is 2.853 with this decision set. Note in the last line of the table, the split in the bounds indicates the heuristic was implemented and the solution may not be optimal. Examining the abbreviated output of the Marginal Analysis program, Table 3.4 shows that the budget was driven negative ten iterations from the end. Entering the same parameters and budget quantity for both starting conditions of the Funnel program produces identical results shown in table XII. Comparing the results in tables XI and XII it may be seen that the return and decision set for the problem are the same for both programs. Interestingly, the Marginal Analysis program was able to correctly select ten items after the heuristic implementation to match the optimal solution produced by the Funnel program. The heuristic employed by Richards and McMasters appears to have been a good choice, although the solution must still be verified.

TABLE X

Input Parameters for a Larger Problem

I	DEMAND	COST	T	ESSEN
1	2.000	0.50	1.90	1.0
2	17.000	3.00	2.20	3.0
3	0.250	3.75	2.50	3.0
4	50.000	5.00	2.60	1.0
5	1.000	10.00	1.60	3.0
6	1.500	12.00	2.50	1.0
7	18.000	15.00	1.50	3.0
8	3.000	20.00	2.50	1.0
9	0.250	25.00	1.50	1.0
10	1.000	25.00	2.00	1.0
11	2.000	25.00	1.50	2.0
12	0.330	50.00	1.50	3.0
13	0.100	50.00	2.00	3.0
14	1.000	61.00	2.00	2.0
15	7.000	80.00	2.30	2.0
16	1.000	91.00	3.70	3.0
17	2.500	150.00	1.75	3.0
18	20.000	210.00	1.70	3.0
19	3.000	450.00	2.70	2.0
20	0.500	500.00	1.75	1.0
21	1.500	710.00	3.00	2.0
22	6.000	1000.00	1.80	1.0
23	2.000	1500.00	2.50	3.0
24	12.000	1500.00	2.00	2.0
25	0.500	10000.00	1.00	1.0

TABLE XI

Marginal Analysis Program Results, Budget=$74825.00

```
*************************************************
*     METHOD: FIXED      BUDGET =    74825.00   *
*************************************************

*************************************************
```

STEP	ITEM	SI	RATIO	BR
1	1	1	2.82236671	74824.50
2	2	1	2.14117622	74821.50
3	2	2	2.08235264	74818.50
4	2	3	2.02352905	74815.50
5	2	4	1.96470547	74812.50
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
440	24	21	0.0004187953	2386.50
441	2	55	0.0004181664	2383.50
442	22	9	0.0003934940	1383.50
443	19	11	0.0003861645	933.50
444	18	40	0.0003540027	723.50
445	4	154	0.0003520278	718.50
446	24	22	0.0003425654	-781.50
446	12	3	0.0003396191	668.50
447	15	23	0.0003359483	588.50
448	7	39	0.0003278039	573.50
449	8	14	0.0003014780	553.50
450	20	2	0.0002939128	53.50
451	4	155	0.0002806289	48.50
452	2	56	0.0002605470	45.50
453	10	6	0.0002369717	20.50
454	1	12	0.0002271836	20.00
455	4	156	0.0002225503	15.00
456	7	40	0.0002018004	0.0

REALIZED PERFORMANCE =   2.8532
BOUNDS (2.9796,   2.6762)

TABLE XII

Funnel Program Results, Budget=$74825.00

THE OBJECTIVE IS MSRT
CONSTRAINT ON COST

THE PROBLEM IS TO MINIMIZE A 25 STAGE PROCESS
XN IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=7.48250D+04
AND XN=7.48250D+04
OPTIMAL XN=7.48250D+04 OPTIMAL RETURN=2.85315D+00

N	XN	SN	XN-1
25	7.4825D+04	0.0	7.4825D+04
24	7.4825D+04	2.10D+01	4.3325D+04
23	4.3325D+04	6.00D+C0	3.4325D+04
22	3.4325D+04	9.00D+00	2.5325D+04
21	2.5325D+04	6.00D+00	2.1065D+04
20	2.1065D+04	2.00D+C0	2.0065D+04
19	2.0065D+04	1.10D+01	1.5115D+04
18	1.5115D+04	4.00D+01	6.7150D+03
17	6.7150D+03	8.00D+00	5.5150D+03
16	5.5150D+03	8.00D+00	4.7870D+03
15	4.7870D+03	2.30D+01	2.947CD+03
14	2.9470D+03	5.00D+C0	2.6420D+03
13	2.6420D+03	2.00D+00	2.5420D+03
12	2.5420D+03	3.00D+C0	2.3920D+03
11	2.3920D+03	7.00D+00	2.2170D+03
10	2.2170D+03	6.00D+C0	2.0670D+03
9	2.0670D+03	2.00D+00	2.0170D+03
8	2.0170D+03	1.40D+01	1.7370D+03
7	1.7370D+03	4.00D+01	1.1370D+03
6	1.1370D+03	9.00D+00	1.0290D+03
5	1.0290D+03	6.00D+00	9.6900D+02
4	9.6900D+02	1.56D+02	1.8900D+02
3	1.8900D+02	4.00D+00	1.7400D+02
2	1.7400D+02	5.60D+01	6.0000D+00
1	6.0000D+00	1.20D+01	0.0

V. CONCLUSIONS

This chapter will discuss the advantages and disadvantages of the two primary solution techniques examined in this paper. Various starting modes of the Funnel program will be looked at in an effort to develop an efficient approach to this type of provisioning problem.

The Marginal analysis program provides a quick, simple and effective approach for generating a solution to the previously defined provisioning problem. However, the solution is not guaranteed to be optimal because the program may implement the Richards heuristic to consume all available resources. The Funnel program provides the optimal solution to the same problem though it takes more computer time to operate. The DP variant has the capability to start the problem from two different conditions, 1) no prior knowledge and 2) prior knowledge or an estimate of the solution. For Case 2, utilizing the ENTER function of the Funnel program allows the rapid construction of the input data file used by DP6 to examine the Xi* values and manipulate the window boundaries as necessary. If the entered solution set is not optimal, the Funnel program will drive the decisions to the optimal decision set in subsequent iterations. Using the timing function mentioned in Chap. 3 a rough approximation of run time is available for each iteration of the program. For Case 1, no prior knowledge, the total estimated time for the $74825.00 problem is nearly 3000 seconds. For Case 2, prior knowledge, a near optimal starting solution for the same problem is driven to the optimal solution in approximately 300 seconds. This 90% savings on computation time makes it clear that an approximation method as an input to the Funnel program would be a better approach to this type

of problem. It should be noted that the run time for the case 2 start is related to the accuracy of the entered decision set. In this instance the entered decision set was specifically constructed to force the Funnel program to adjust the window boundaries in all stages at least once. A better solution than the one entered could produce a much shorter run time.

There are tradeoffs in the Funnel Program for computational efficiency. Variations of the multiplier that creates the window can drive the individual program run time directly. A window that is too large wastes computation time on resource values that are not needed. If the window width is too small, multiple iterations of the program may be required to compensate for the increased number of boundary violations in the effort to reallocate the resource consumed at the various stages. For the verification of a decision set provided by the Marginal Analysis program the multiplier could be set to 1.0 providing a very narrow window of resource to be examined. Due to the narrow width of the window, if the Xi* value changes in any stage the boundary will be recorded and displayed to the user as violated. This notification would indicate that the entered solution is not optimal for the problem.

The computer time expended for the verification alternative would be quite low even for large budget problems. In the 25-stage problem, for example, approximately one second is required to verify the correctness of the optimal solution if it is entered initially. If the verification alternative indicates the decision is not optimal, the time required to drive the entered solution to the optimal solution is controlled by the magnitude of the changes in the decision set and the number of iterations required to re-allocate the resources throughout all the stages. The user must make a value judgement on the importance of the optimal solution versus the additional computer costs.

In conclusion, the emphasis of this paper is to create an algorithm to determine the optimal solution in a quick and easy manner. It is evident that a combination of the Marginal Analysis and Funnel programs is an efficient approach for solving this type of problem. Using the Marginal Analysis program to generate a best estimate of the solution, the ENTER function with the Funnel program can then be applied to confirm the optimal solution or to create an input data file that DP6 may use to find the optimal solution. The provisioning problems faced by NAVSUP, as described in Chap 1, may be solved with this method if they can be formatted in the recursive DP construct. The program DP6 in particular solves only the minimization of MSRT with a constraint on cost. Variations on this program could be devised to solve additional objectives like supply material availability(SMA), time weighted units short(TWUS), etc., by changing the computation procedures in the subroutines. The constraint however, is currently restricted to the linear form,

$$\sum Wi*Si<=B \quad i=1,...,N \tag{5.1}$$

where Wi is resources required per unit, Si is the number of units of item i, and B is the total resource available.

APPENDIX A
DP6 CODE

This appendix contains the computer listing of the main program and subroutines for the funneling procedure. Some of the subroutines contain code for objectives and constraints other than those discussed in this thesis.

```
C
C     DATA DESCRIPTION FOR CARD NUMBER 1
C     --------.---------.---------.-------------------------------------
C     COLUMNS . JUSTIFY . VARIBLE . MEANING
C             .         . NAME    .
C     --------.---------.---------.-------------------------------------
C     1 - 5   . COL 5   . NMAX    . THE NUMBER OF STAGES IN THE PROBLEM
C     --------.---------.---------.-------------------------------------
C     6 - 10  . COL 10  . MTAPE   . LOGICAL TAPE NO OF THE MAIN TAPE
C             .         .         . IF THIS IS LEFT BLANK, THE COMPUTER
C             .         .         . WILL USE STANDARD SCRATCH TAPE 3
C     --------.---------.---------.-------------------------------------
C     11 - 15 . COL 15  . SOLVE   . =  OLD    IF THE OPTIMAL DECISION
C             .         .         .           FUNCTIONS HAVE ALREADY BEEN
C             .         .         .           CALCULATED AND ARE LOADED
C             .         .         .           ON LOGICAL TAPE MTAPE
C             .         .         .-------------------------------------
C             .         .         . =  NEW    IF THE OPTIMAL DECISION
C             .         .         .           FUNCTIONS MUST BE
C             .         .         .           CALCULATED AND STORED ON
C             .         .         .           LOGICAL TAPE MTAPE BEFORE
C             .         .         .           SOLVING THE PROBLEM
C     --------.---------.---------.-------------------------------------
C
C
C
C     DATA DESCRIPTION FOR THE NEXT GROUP OF CARDS
C     IF SOLVE = NEW  ON CARD NUMBER 1, THEN EACH OF THE NMAX STAGES
C     MUST BE DECRIBED BY THE CARDS DISCUSSED BELOW. YOU CAN USE 1 CARD
C     PER STAGE OR YOU CAN MAKE 1 CARD DESCRIBE MANY ADJACENT STAGES IF
C     THEY ARE SIMILAR. THE STAGES MUST BE DESCRIBED IN NUMERICAL ORDER
C     STARTING WITH STAGE 1 (I.E. STAGE 1, THEN 2, ...., THEN NMAX)
C     OMIT THIS PACK OF CARDS WHICH DESCRIBE THE STAGES IF SOLVE = OLD
C     --------.---------.---------.-------------------------------------
C     COLUMNS . JUSTIFY . VARIBLE . MEANING
C             .         . NAME    .
C     --------.---------.---------.-------------------------------------
C     1 - 5   . COL 5   . NSTAGE  . LOWEST NUMBERED STAGE FOR WHICH
C             .         .         . THIS CARD APPLIES
C     --------.---------.---------.-------------------------------------
C     6 - 10  . COL 10  . NDITTO  . HIGHEST NUMBERED STAGE FOR WHICH
C             .         .         . THIS CARD APPLIES. IF THESE COLUMNS
C             .         .         . ARE LEFT BLANK, THEN NDITTO WILL BE
C             .         .         . TAKEN AS EQUAL TO NSTAGE
C     --------.---------.---------.-------------------------------------
C     11 - 20 . ANY     . XLOW    . LOWEST VALUE OF XN FOR STAGE NSTAGE
C     --------.---------.---------.-------------------------------------
C     21 - 30 . ANY     . XHIGH   . HIGHEST VALUE OF XN FOR STAGE NSTAGE
```

```
C     ----------.----------.----------.---------------------------------DP600490
C     31 - 40 . ANY      . DELX     . INCREMENT IN XN FOR STAGE NSTAGE    DP600500
C     ----------.----------.----------.---------------------------------DP600510
C     41 - 46 . COL 46   . XMODE    . =   MIN  IF STAGE NSTAGE IS TO BE   DP600520
C             .          .          .          MINIMIZED                  DP600530
C             .          .          .-----------------------------------DP600540
C             .          .          . =   MAX  IF STAGE NSTAGE IS TO BE   DP600550
C             .          .          .          MAXIMIZED                  DP600560
C     ----------.----------.----------.---------------------------------DP600570
C     47 - 52 . COL 52   . XSTAGE   . =   SUM  IF THE COMPOSITION         DP600580
C             .          .          .          OPERATOR BETWEEN STAGES    DP600590
C             .          .          .          NSTAGE AND NSTAGE-1 IS     DP600600
C             .          .          .          ADDITION                   DP600610
C             .          .          .-----------------------------------DP600620
C             .          .          . =  MULT  IF THE COMPOSITION         DP600630
C             .          .          .          OPERATOR BETWEEN STAGES    DP600640
C             .          .          .          NSTAGE AND NSTAGE-1 IS     DP600650
C             .          .          .          MULTIPLICATION             DP600660
C             .          .          .-----------------------------------DP600670
C             .          .          . =MINMAX  IF THE COMPOSITION         DP600680
C             .          .          .          OPERATOR BETWEEN STAGES    DP600690
C             .          .          .          NSTAGE AND NSTAGE-1 IS TO  DP600700
C             .          .          .          MINIMIZE THE MAXIMUM       DP600710
C             .          .          .          INDIVIDUAL STAGE RETURN    DP600720
C             .          .          .-----------------------------------DP600730
C             .          .          . =MAXMIN  IF THE COMPOSITICN         DP600740
C             .          .          .          OPERATOR BETWEEN STAGES    DP600750
C             .          .          .          NSTAGE AND NSTAGE-1 IS TO  DP600760
C             .          .          .          MAXIMIZE THE MINIMUM       DP600770
C             .          .          .          INDIVIDUAL STAGE RETURN    DP600780
C             .          .          .-----------------------------------DP600790
C             .          .          . =        THIS VARIAELE IS NOT USED  DP600800
C             .          .          .          WHEN NSTAGE=1, THEREFORE   DP600810
C             .          .          .          COLUMNS 47 TO 52 ARE       DP600820
C             .          .          .          IGNORED WHEN NSTAGE=1 AND  DP600830
C             .          .          .          MAY BE LEFT ELANK          DP600840
C             .          .          .          HOWEVER IF NDITTO IS       DP600850
C             .          .          .          GREATER THAN 1, THEN XSTAGEDP600860
C             .          .          .          MUST BE SPECIFIED IN       DP600870
C             .          .          .          ACCORDANCE WITH THE        DP600880
C             .          .          .          COMPOSITON OPERATOR FOR THEDP600890
C             .          .          .          REST OF STAGES DESCRIBED ONDP600900
C             .          .          .          THIS CARD                  DP600910
C     ----------.----------.----------.---------------------------------DP600920
C     53 - 55 . COL 55   . MODES    . =  1     IF THE FIBONACCI SEARCH IS DP600930
C             .          .          .          TO BE USED                 DP600940
C             .          .          .-----------------------------------DP600950
C             .          .          . =  2     IF THE FIBONACCI SEARCH IS DP600960
```

```
C                                               NOT TO BE USED              DP600970
C     --------.----------.----------.----------------------------------------DP600980
C     56 - 64 .   ANY    . XMIN1L   . THIS VARIABLE IS THE LOWEST BOUNDARYDP600990
C             .          .          . VALUE ANTICIPATED FOR THIS STAGE    DP601000
C     --------.----------.----------.----------------------------------------DP601010
C     65 - 73 .   ANY    . XMIN1H   . THIS VARIABLE IS THE HIGHEST BOUNDARDP601020
C             .          .          . VALUE ANTICIPATED FOR THIS STAGE    DP601030
C     --------.----------.----------.----------------------------------------DP601040
C                                                                         DP601050
C                                                                         DP601060
C                                                                         DP601070
C     DATA DESCRIPTICN OF LAST CARD FOR PROBLEM                           DP601080
C     THE OPTIMUM DECISIONS ARE TO BE PRINTED OUT FOR A PROCESS WITH      DP601090
C     NMAX STAGES, FURTHER, FN(XN) IS TO BE OPTIMIZED WITH RESPECT TO XNDP601100
C     WHEN THE PROGRAM COMPLETES THE REQUESTED PRINTOUT IT WILL RETURN    DP601110
C     TO THE BEGINNING AND START A NEW PROBLEM IF THERE IS DATA, IF NO    DP601120
C     DATA, IT WILL EXIT                                                  DP601130
C     --------.----------.----------.----------------------------------------DP601140
C     COLUMNS . JUSTIFY  . VARIBLE  . MEANING                             DP601150
C             .          . NAME     .                                     DP601160
C     --------.----------.----------.----------------------------------------DP601170
C     1 - 15  . ANY      . XN1      . AT STAGE NMAX, OPTIMIZE FN(XN) FOR  DP601180
C             .          .          . XN NOT LESS THAN XN1                DP601190
C     --------.----------.----------.----------------------------------------DP601200
C     16 - 30 . ANY      . XN2      . AT STAGE NMAX, OPTIMIZE FN(XN) FOR  DP601210
C             .          .          . XN NOT GREATER THAN XN2             DP601220
C     --------.----------.----------.----------------------------------------DP601230
C                                                                         DP601240
C                                                                         DP601250
C                                                                         DP601260
C     DESCRIPTION OF THE SUBROUTINE TRANFM                                DP601270
C     THE MAIN PROGRAM TRANSMITS XN,DN, AND NUMBER (THE SEQUENCE NUMBER DP601280
C     OF THE PROBLEM) TO THE SUBROUTINE AND THE SUBROUTINE CALCULATES     DP601290
C     YN (THE OUTPUT CF THE STAGE I.E. YN=XN-1)                           DP601300
C                                                                         DP601310
C                                                                         DP601320
C                                                                         DP601330
C     DESCRIPTION OF THE SUBROUTINE STGRET                                DP601340
C     THE MAIN PROGRAM TRANSMITS XN,DN,YN, AND NUMBER TO THE SUBROUTINE,DP601350
C     AND THE SUBROUTINE CALCULATES RN (THE STAGE RETURN)                 DP601360
C                                                                         DP601370
C                                                                         DP601380
C                                                                         DP601390
C     DESCRIPTION OF SUBROUTINE STORE                                     DP601400
C     THE MAIN PROGRAM CALLS THE SUBROUTINE RIGHT AFTER READING THE       DP601410
C     FIRST DATA CARD AND DEFINING NUMBER. THE PURPOSE OF THIS            DP601420
C     SUBROUTINE IS TO ALLOW THE STORING OF CONSTANTS IN COMMON STORAGE DP601430
C     FOR THE POSSIBLE USE OF SUBROUTINES STGRET AND TRANFM               DP601440
```

```
C                                                                       DP601450
C     DATA WHICH WILL BE READ FROM DATA CARDS AT OBJECT TIME BY         DP601460
C     THE SUBROUTINE STORE MUST BE INSERTED BETWEEN THE FIRST DATA CARD DP601470
C     AND THE GROUP OF CARDS WHICH DESCRIBE THE INDIVIDUAL STAGES       DP601480
C                                                                       DP601490
C                                                                       DP601500
C                                                                       DP601510
C     DESCRIPTION OF SUBROUTINE DLIMIT                                  DP601520
C     THE MAIN PROGRAM TRANSMITS N,XN, AND NUMBER TO THE SUBROUTINE,    DP601530
C     AND THE SUBROUTINE CALCULATES DLOW, DHIGH, AND DELD               DP601540
C     WHERE DLOW = LOWEST VALUE OF DN FOR THAT PARTICULAR VALUE OF N ANDDP601550
C                  XN                                                   DP601560
C          DHIGH = HIGHEST VALUE OF DN  FOR THAT PARTICULAR VALUE OF N  DP601570
C                  AND XN                                               DP601580
C           DELD = INCREMENT IN DN FOR THAT PARTICULAR VALUE OF N AND XNDP601590
C     IF ANY OF THESE VARIBLES DO NOT CHANGE, THEY CAN BE SET IN        DP601600
C     SUBROUTINE STORE AND NOT MENTIONED IN SUBROUTINE DLIMIT           DP601610
C                                                                       DP601620
C                                                                       DP601630
C                                                                       DP601640
C     IF ANY SUBROUTINE SETS NUFF TO -1  THEN THE PROGRAM IGNORES THE   DP601650
C     VALUE OF DN BEING PROCESSED AT THAT MOMENT, AND IGNORES ALL LARGERDP601660
C     VALUES OF DN FOR THAT PARTICULAR VALUE OF XN AT THAT PARTICULAR   DP601670
C     STAGE--THIS FEATURE IS DISABLED WHEN USING THE FIBONACCI SEARCH   DP601680
C                                                                       DP601690
C                                                                       DP601700
C                                                                       DP601710
C     DESCRIPTION OF THE SUBROUTINE ENTER                               DP601720
C     THIS SUBROUTINE READS A DECISION SET FROM A FILE DESIGNATED BY THEDP601730
C     AND FORMS AN INPUT DATA FILE THAT IS COMPATIBLE WITH THE ADJUST SUDP601740
C     TINE FOR BOUNDARY MANIPULATION.  IT MAY ALSO BE USED FOR VERIFICATDP601750
C     AN OPTIMAL SOLUTION BY SETTING EQUAL TO ONE THE CONSTANT MULTIPLIEDP601760
C     CONTROLS THE WINDOW WIDTH.                                        DP601770
C                                                                       DP601780
C                                                                       DP601790
C                                                                       DP601800
C     DESCRIPTION OF THE SUBROUTINE ADJUST                              DP601810
C     THIS SUBROUTINE MANIPULATES THE UPPER AND LOWER BOUNDARIES OF THE DP601820
C     COMPUTATIONAL WINDOWS TO ALLOW THE DP PROCESS TO LOCATE THE       DP601830
C     OPTIMAL XI AND SI VALUES.                                         DP601840
C                                                                       DP601850
C                                                                       DP601860
C                                                                       DP601870
C     THIS PROGRAM USES TAPES 3 AND 4 FOR SCRATCH                       DP601880
C     NSS2 ZERO FOR ENTIRE PROBLEM SOLVING AND OUTPUT                   DP601890
C     NSS2 NON-ZERO FOR OUTPUT FROM PREVIOUS PROBLEM                    DP601900
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)DP601910
      DIMENSION D(101),E(101),F(101),G(101),BOUND(101),V(101)           DP601920
```

```
      REAL * 8 HOLMIN/'   MIN'/, HOLOLD/'  OLD'/                       DP601930
      REAL * 8 XMODE, XSTAGE, SOLVE                                     DP601940
      REAL * 8 TAB(4)                                                   DP601950
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF                         DP601960
      COMMON /CON/TAB                                                   DP601970
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER       DP601980
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM              DP601990
      CCMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3                   DP602000
      COMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2                           DP602010
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))                             DP602020
      CALL SETIME                                                       DP602030
      XMIN1L=0.0                                                        DP602040
      XMIN1H=0.0                                                        DP602050
      LOUTPE=6                                                          DP602060
      INTAPE=7                                                          DP602070
      KEEP=9                                                            DP602080
      NEXT1=10                                                          DP602090
      IBALL=11                                                          DP602100
      NOUTPE=12                                                         DP602110
      WRITE(NOUTPE,1000) HOLMIN, HOLOLD, (TAB(I), I=1,4)                DP602120
 1000 FORMAT(1X, 6A6)                                                   DP602130
      MAX=10001                                                         DP602140
      CALL SEARCH(0.0,0.0,-1.0,0.0,DNBEST,BEST)                         DP602150
      NPAGE=0                                                           DP602160
      NUMBER=0                                                          DP602170
  240 READ(INTAPE,100,END=2000) NMAX,MTAPE,SOLVE                        DP602180
    6 WRITE(KEEP,100) NMAX,MTAPE,SOLVE                                  DP602190
  100 FORMAT(2I5,A5)                                                    DP602200
      MTAP = MTAPE                                                      DP602210
      NDITTC=0                                                          DP602220
      NCALC=4                                                           DP602230
      IF(MTAPE-4) 405,501,405                                           DP602240
  501 NCALC=3                                                           DP602250
      GO TO 172                                                         DP602260
  405 IF(MTAPE) 322,322,172                                             DP602270
  322 MTAPE=3                                                           DP602280
  172 NTAPE=MTAPE                                                       DP602290
      REWIND MTAPE                                                      DP602300
      NUMBER=NUMBER+1                                                   DP602310
C     CALL STORE WAS MOVED FROM HERE                                    DP602320
      IF(SOLVE-HOLOLD) 301,302,301                                      DP602330
  302 NSS2=1                                                            DP602340
      NPAGE=NPAGE+1                                                     DP602350
C     WRITE(NOUTPE,152) NPAGE                                           DP602360
C 152 FORMAT(1H160X4HPAGEI4)                                            DP602370
C     WRITE(NOUTPE,303)                                                 DP602380
C     WRITE(NOUTPE,1303) MTAPE                                          DP602390
C 303 FORMAT (53H THE TABLES OF FN(XN) AND DN(XN) WHICH ARE ON LOGICAL) DP602400
```

```
C1303 FORMAT(43H TAPEI5, WILL BE USED TO SOLVE THIS PROBLEM)                  DP602410
      READ(MTAPE) NMAX                                                        DP602420
      REWIND MTAPE                                                            DP602430
      GO TO 304                                                               DP602440
  301 NSS2=0                                                                  DP602450
      CALL STORE                                                              DP602460
      NPAGE=NPAGE+1                                                           DP602470
C     WRITE(NOUTPE,152) NPAGE                                                 DP602480
C     WRITE(NOUTPE,305)                                                       DP602490
C     WRITE(NOUTPE,1305) MTAPE                                                DP602500
C 305 FORMAT(58H THE COMPUTER IS TO CALCULATE TABLES OF FN(XN) AND DN(XNDP602510
C    1),)                                                                     DP602520
C1305 FORMAT(31H AND STORE THEM ON LOGICAL TAPEI5)                            DP602530
      REWIND NCALC                                                            DP602540
  500 WRITE(MTAPE) NMAX                                                       DP602550
C 304 WRITE(NOUTPE,306) NMAX                                                 DP602560
C  CHANGED TO CONTINUE FOR ABBREVIATED PRINTOUT                               DP602570
  304 CONTINUE                                                                DP602580
C 306 FORMAT(16H THE PROBLEM HASI5,7H STAGES)                                 DP602590
  314 IF(NSS2) 197,109,197                                                    DP602600
  109 LINES = 6                                                               DP602610
C     WRITE(NOUTPE,2006) NMAX, MTAP, SOLVE                                    DP602620
C2006 FORMAT(/// ' DATA CARD  ', 2I7, A7)                                     DP602630
  121 DO 123 N=1,NMAX                                                         DP602640
      NM1=N-1                                                                 DP602650
      IF(NDITTO-N) 241,132,132                                                DP602660
  241 READ(INTAPE,124) NSTAGE,NDITTO,XLOW,XHIGH,DELX,XMODE,XSTAGE,MODES DP602670
     1,XMIN1L,XMIN1H                                                          DP602680
  124 FORMAT(2I5,3E10.8,2A6,I3,2E9.2)                                         DP602690
C     WRITE(NOUTPE,2007) NSTAGE, NDITTO, XLOW, XHIGH, DELX, XMODE,            DP602700
C    1         XSTAGE, MODES                                                  DP602710
C     WRITE(NOUTPE,2009)XMIN1L,XMIN1H                                         DP602720
C2009 FORMAT(' XN-1 LOW,HIGH ARE:',2E9.2)                                     DP602730
C2007 FORMAT(/// ' DATA CARD  ', 2I5, 2X, 3F10.3, 2A6, I3)                    DP602740
      IF(LINES-41) 30,30,31                                                   DP602750
   31 NPAGE=NPAGE+1                                                           DP602760
C     WRITE(NOUTPE,152) NPAGE                                                 DP602770
      LINES=1                                                                 DP602780
   30 LINES = LINES + 9                                                       DP602790
      ITOP=(XHIGH-XLOW)/DELX+1.001                                            DP602800
      IF(MODES-1) 600,601,600                                                 DP602810
  600 MODES=-1                                                                DP602820
  601 IF(NSTAGE-N) 125,245,125                                                DP602830
  125 WRITE(NOUTPE,127)                                                       DP602840
  127 FORMAT(23H1DATA CARD OUT OF ORDER)                                      DP602850
      GO TO 2003                                                              DP602860
  245 IF(NSTAGE-NDITTO) 246,126,126                                           DP602870
  246 WRITE(NOUTPE,247) N                                                     DP602880
```

```
      WRITE(NOUTPE,1247) NSTAGE,NDITTO                                  DP602890
  247 FORMAT(///46H THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE,I5)   DP602900
 1247 FORMAT(17H APPLIES TO STAGE,I5,11H THRU STAGE,I5,10H INCLUSIVE)   DP602910
      LINES=LINES+4                                                     DP602920
      XXMODE = XMODE - HOLMIN                                           DP602930
C     WRITE(6,1001) XXMODE                                              DP602940
C1001 FORMAT(1X, A6)                                                    DP602950
  126 IF(XMODE-HOLMIN) 230,231,230                                      DP602960
  231 XMODE=1.0                                                         DP602970
C     WRITE(NOUTPE,232) N                                               DP602980
C 232 FORMAT(//6H STAGEI5,18H IS A MINIMIZATION)                        DP602990
      GO TO 520                                                         DP603000
  230 XMODE=-1.0                                                        DP603010
C     WRITE(NOUTPE,234) N                                               DP603020
C 234 FORMAT(//6H STAGEI5,18H IS A MAXIMIZATION)                        DP603030
  520 DO 129 I=1,4                                                      DP603040
      IF(XSTAGE-TAB(I)) 129,130,129                                     DP603050
  129 CONTINUE                                                          DP603060
  521 NNFLAG=5                                                          DP603070
      GO TO 502                                                         DP603080
  130 NNFLAG=I                                                          DP603090
CADDED HERE TO COMPENSATE FOR ABBREVIATED OUTPUT                        DP603100
      GO TO 132                                                         DP603110
C CHANGED TO CONTINUE FOR ABBREVIATED OUTPUT                            DP603120
  502 CONTINUE                                                          DP603130
C 502 IF(ITOP-MAX) 316,316,112                                          DP603140
C 316 WRITE(NOUTPE,110)                                                 DP603150
C 317 WRITE(NOUTPE,1110) XLOW,XHIGH,DELX                                DP603160
C 110 FORMAT(54H THE STATE VARIBLE IS BEING TREATED AS DISCRETE, GOING) DP603170
C1110 FORMAT(9H FROM XN=E14.5,7H TO XN=E14.5,13H IN STEPS OF E14.5)     DP603180
C1502 IF(MODES) 318,318,321                                             DP603190
C 318 WRITE(NOUTPE,510)                                                 DP603200
C 510 FORMAT(41H TOTAL ENUMERATION IS USED FOR THIS STAGE)              DP603210
C     GO TO 320                                                         DP603220
C 321 WRITE(NOUTPE,511)                                                 DP603230
C 511 FORMAT(40H FIBONACCI SEARCH IS USED FOR THIS STAGE)               DP603240
C 320 GO TO (11,12,13,14,132),NNFLAG                                    DP603250
C  11 WRITE(NOUTPE,16) NM1,N                                            DP603260
C     WRITE(NOUTPE,1016)                                                DP603270
C  16 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6)    DP603280
C1016 FORMAT(12H IS ADDITION)                                           DP603290
C     GO TO 132                                                         DP603300
C  12 WRITE(NOUTPE,17) NM1,N                                            DP603310
C     WRITE(NOUTPE,1017)                                                DP603320
C  17 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6)    DP603330
C1017 FORMAT(18H IS MULTIPLICATION)                                     DP603340
C     GO TO 132                                                         DP603350
C  13 WRITE(NOUTPE,18) NM1,N                                            DP603360
```

```
C        WRITE(NOUTPE,1018)                                                    DP603370
C   18   FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6)        DP603380
C1018    FORMAT(34H IS TO MAXIMIZE THE MINIMUM RETURN)                         DP603390
C        GO TO 132                                                             DP603400
C   14   WRITE(NOUTPE,19) NM1,N                                                DP603410
C        WRITE(NOUTPE,1019)                                                    DP603420
C   19   FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6)        DP603430
C1019    FORMAT(34H IS TO MINIMIZE THE MAXIMUM RETURN)                         DP603440
  132    DO 136 I=1,ITOP                                                       DP603450
  134    BEST=1.0E+35*XMODE                                                    DP603460
         X=I-1                                                                 DP603470
         XN=XLOW+X*DELX                                                        DP603480
         NUFF=1                                                                DP603490
         CALL DLIMIT                                                           DP603500
         IF(MODES) 602,602,603                                                 DP603510
  603    CALL SEARCH(DLOW,DHIGH,DELD,XMODE,DNBEST,BEST)                        DP603520
         GO TO 400                                                             DP603530
  602    KTOP=(DHIGH-DLOW)/DELD+1.001                                          DP603540
C        NEXT LINE INSERTED FOR TESTING 1 AUG 84                               DP603550
         RNBST=999.                                                            DP603560
  223    DO 137 J=1,KTOP                                                       DP603570
         X=J-1                                                                 DP603580
         DN=DLOW+X*DELD                                                        DP603590
         CALL TRANFM                                                           DP603600
         CALL STGRET                                                           DP603610
         IF(NUFF) 400,401,401                                                  DP603620
  401    IF(N-1) 504,504,503                                                   DP603630
  504    CN=RN                                                                 DP603640
         GO TO 143                                                             DP603650
  503    K=(YN-YLOW)/DELY+1.001                                                DP603660
         IF(K) 212,212,209                                                     DP603670
  212    WRITE(NOUTPE,210) N,XN,DN                                             DP603680
         WRITE(NOUTPE,1210) YN                                                 DP603690
  210    FORMAT(///9H AT STAGEI5,9H WITH XN=E15.8,8H AND DN=E15.8)             DP603700
 1210    FORMAT(12H XN-1 EQUALS,E16.8,21H AND IS OUT OF LIMITS///)             DP603710
C        WRITE(NOUTPE,5)XMIN1L,XMIN1H                                          DP603720
C5       FORMAT(5X,2F10.4)                                                     DP603730
         GO TO 2003                                                            DP603740
  209    IF(K-JTOP) 205,206,212                                                DP603750
  206    RNM1=FNM1(JTOP)                                                       DP603760
         GO TO 207                                                             DP603770
  205    X=K-1                                                                 DP603780
         X=YLOW+X*DELY                                                         DP603790
         RNM1=FNM1(K)+(FNM1(K+1)-FNM1(K))*(YN-X)/DELY                          DP603800
  207    GO TO (21,22,23,24,248),NNFLAG                                        DP603810
  248    WRITE(NOUTPE,249) N                                                   DP603820
  249    FORMAT(43H1COMPOSITION OPERATOR NOT DEFINED FOR STAGE,I5)             DP603830
         GO TO 2003                                                            DP603840
```

```
   21 QN=RN+RNM1                                                          DP60385
      GO TO 143                                                           DP603860
   22 QN=RN*RNM1                                                          DP603870
      GO TO 143                                                           DP603880
   23 IF(RN-RNM1) 141,141,142                                             DP603890
  141 QN=RN                                                               DP603900
      GO TO 143                                                           DP603910
  142 QN=RNM1                                                             DP603920
      GO TO 143                                                           DP603930
   24 IF(RN-RNM1) 145,145,146                                             DP603940
  145 QN=RNM1                                                             DP603950
      GO TO 143                                                           DP603960
  146 QN=RN                                                               DP603970
  143 IF((QN-BEST)*XMODE) 144,137,137                                     DP603980
  144 BEST=QN                                                             DP603990
      DNBEST=DN                                                           DP604000
C     NEXT LINE INSERTED FOR TESTING 1AUG84                               DP604010
      RNBST=RN                                                            DP604020
C     WRITE(IBALL,5) DN,BEST                                              DP604030
C5    FORMAT(' AT',F10.2,' THE TOTAL RETURN IS',F15.9)                    DP604040
  137 CONTINUE                                                            DP604050
  400 WRITE(NCALC) BEST,DNBEST                                            DP604060
      BEST1=RNBST*365.                                                    DP604070
      BEST2=SUMELT*BEST1/(E(N)*A(N)*B(N))                                 DP604080
C     WRITE(IBALL,2008) N,XN,DNBEST,RNBST,BEST1,BEST2                     DP604090
C2008 FORMAT(I5,2F10.2,3F15.10)                                           DP604100
  136 CONTINUE                                                            DP604110
      REWIND NCALC                                                        DP604120
      WRITE(NTAPE) XLOW,XHIGH,DELX,ITOP,XMODE,XMIN1L,XMIN1H               DP604130
      DO 404 I=1,ITOP                                                     DP604140
  404 READ(NCALC) DUM,DNOFXN(I)                                           DP604150
      REWIND NCALC                                                        DP604160
      WRITE(NTAPE) (DNOFXN(II),II=1,ITOP)                                 DP604170
      DO 402 I=1,ITOP                                                     DP604180
  402 READ(NCALC) FN(I),DUM                                               DP604190
      REWIND NCALC                                                        DP604200
      YLOW=XLOW                                                           DP604210
      YHIGH=XHIGH                                                         DP604220
      DELY=DELX                                                           DP604230
      JTOP=ITOP                                                           DP604240
  123 CONTINUE                                                            DP604250
      WRITE(NTAPE) (FN(II),II=1,ITOP)                                     DP604260
  161 END FILE MTAPE                                                      DP604270
  197 REWIND MTAPE                                                        DP604280
      READ(MTAPE) NMAX                                                    DP604290
      N=NMAX                                                              DP604300
      J=N-1                                                               DP604310
      IF(J) 407,407,201                                                   DP604320
```

```
  201 DO 522 I=1,J
      READ(MTAPE)
  522 READ(MTAPE)
  407 READ(MTAPE)  XLOW,XHIGH,DELX,ITOP,XMODE
      READ(MTAPE)
      READ(MTAPE)  (FN(I),I=1,ITOP)
      BACKSPACE MTAPE
      BACKSPACE MTAPE
      READ(INTAPE,162)  XN1,XN2
  162 FORMAT(2E15.8)
      NPAGE=NPAGE+1
C     WRITE(NOUTPE,152)  NPAGE
C     WRITE(NOUTPE,2008)  XN1,  XN2
C2008 FORMAT(/// ' DATA CARD  ',  2F20.8 ///)
C     IF(XMODE)  323,323,324
  324 WRITE(NOUTPE,325)  N
  325 FORMAT(29H THE PROBLEM IS TO MINIMIZE AI6,14H STAGE PROCESS)
      GO TO 327
  323 WRITE(NOUTPE,326)  N
  326 FORMAT(29H THE PROBLEM IS TO MAXIMIZE AI6,14H STAGE PROCESS)
  327 WRITE(NOUTPE,328)  XN1
      WRITE(NOUTPE,1328)  XN2
  328 FORMAT (42H XN IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=1PE14.5)
 1328 FORMAT(8H  AND XN=1PE14.5)
  171 IF(XLOW-XN1)  175,175,176
  176 WRITE(NOUTPE,177)  XLOW,XHIGH
      WRITE(NOUTPE,1177)  XLOW,XHIGH
  177 FORMAT(47H0THE PROGRAM ONLY HAS INFORMATION ON XN BETWEENE16.8)
 1177 FORMAT(4H ANDE16.8)
      GO TO 240
  175 IF(XN2-XHIGH)  181,181,176
  181 IF(XN1-XN2)  178,178,176
  178 IX1=(XN1-XLOW)/DELX+1.001
      IX2=(XN2-XLOW)/DELX+1.001
      BEST=1.0E+35*XMODE
      DO 182 J=IX1,IX2
      IF((FN(J)-BEST)*XMODE)  183,182,182
  183 BEST=FN(J)
      JSAVE=J
  182 CONTINUE
      X=JSAVE-1
      XN=XLOW+X*DELX
      BEST=BEST*365.0
      WRITE(NOUTPE,184)  XN,BEST
  184 FORMAT(12H OPTIMAL XN=1PE14.5,16H OPTIMAL RETURN=1PE14.5)
      READ(MTAPE)  (DNOFXN(II),II=1,ITOP)
  186 DN=DNOFXN(JSAVE)
      CALL TRANFM
```

```
  204 WRITE(NOUTPE,188)                                                    DP604810
  188 FORMAT(7H0       N18X2HXN18X2HDN16X4HXN-1)                           DP604820
      NN=0                                                                 DP604830
  203 WRITE(NOUTPE,189) N,XN,DN,YN                                         DP604840
 1203 WRITE(KEEP,190) N,DN, XN,C(N),XMIN1L,XMIN1H                          DP604850
  189 FORMAT(I7, 1P3E20.8)                                                 DP604860
  190 FORMAT(1I5,5F10.2)                                                   DP604870
      NN=NN+1                                                              DP604880
      N=N-1                                                                DP604890
C     IF((N-1).GE.0.0)GO TO 1243                                          DP604900
C1244       CALL ADJUST                                                    DP604910
 1243 IF(N-1) 244,193,193                                                  DP604920
C   MOVED 244 LABEL UP ONE LINE SHOULD BE ON GO TO 240                     DP604930
  244 IF(KODE3.EQ.0) GO TO 1333                                            DP604940
      CALL ENTER2                                                          DP604950
 1333 CONTINUE                                                             DP604960
      CALL SADJUS                                                          DP604970
      CALL GETIME (IET)                                                    DP604980
      ET=IET*.000026                                                       DP604990
      WRITE(NOUTPE,2) ET                                                   DP605000
      WRITE(LOUTPE,2) ET                                                   DP605010
    2 FORMAT(' ELAPSED TIME FOR EXECUTION WAS ',F16.6,' SECONDS.')         DP605020
C     CALL FRTCMS(COM(1),COM(3),COM(J5),COM(6),COM(1),COM(2),COM(I5),      DP605030
C    1COM(6))                                                              DP605040
      GO TO 240                                                            DP605050
  193 DO 406 I=1,4                                                         DP605060
  406 BACKSPACE MTAPE                                                      DP605070
      READ(MTAPE) XLOW,XHIGH,DELX,ITOP,XMODE,XMIN1L,XMIN1H                 DP605080
      READ(MTAPE) (DNOFXN(II),II=1,ITOP)                                   DP605090
      L=(YN-XLOW)/DELX+1.001                                               DP605100
      IF(L) 214,214,215                                                    DP605110
  214 L=1                                                                  DP605120
  219 NP1=N+1                                                              DP605130
      WRITE(NOUTPE,210) NP1,XN,DN                                          DP605140
      WRITE(NOUTPE,1210) YN                                                DP605150
      NN=NN+7                                                              DP605160
  215 IF(L-ITOP) 218,216,217                                               DP605170
  216 DN=DNOFXN(L)                                                         DP605180
      XN=YN                                                                DP605190
      GO TO 196                                                            DP605200
  217 L=ITOP                                                               DP605210
      GO TO 219                                                            DP605220
  218 XN=YN                                                                DP605230
  194 X=I-1                                                                DP605240
      X=XLOW+X*DELX                                                        DP605250
C     WRITE(NOUTPE,9876) N,L,DNOFXN(L),FN(L),DNOFXN(L),XN,X,DELX           DP605260
C9876 FORMAT(2I5,5X,3F5.2,3F10.3)                                          DP605270
      DN=DNOFXN(L)+(DNOFXN(L+1)-DNOFXN(L))*(XN-X)/DELX                     DP605280
```

```
  196 CALL TRANFM
      IF(NN-50) 203,203,220
  220 NPAGE=NPAGE+1
C     WRITE(NOUTPE,152) NPAGE
      GO TO 204
  112 WRITE(NOUTPE,113)
      WRITE(NOUTPE,1113) MAX,ITOP
  113 FORMAT(52H THIS PROGRAM LIMITS THE NUMBER OF DISCRETE STEPS OF)
 1113 FORMAT(21H THE STATE VARIBLE TOI6,26H AND THIS PROBLEM REQUIRESI6)
 2002 FORMAT('1',' ERROR HALT')
 2003 WRITE(NOUTPE,2002)
      STOP
 2000 WRITE(NOUTPE,2001)
 2001 FORMAT('1','   END OF DATA FILE')
      STOP
      END
CSEARCH        DISCRETE FIBONACCI SEARCH SUBROUTINE
      SUBROUTINE SEARCH(AA,BB,DELYY,XXMODE,YYBEST,BEST)
      DIMENSION F(150)
C
C     OPTIMIZE WITH RESPECT TO Y BETWEEN AA AND BB IN STEPS OF DELYY
C     STORE OPTIMUM Y IN YYBEST AND THE OPTIMUM VALUE OF THE OBJECTIVE
C     FUNCTION IN BEST
C     XXMODE=-1 FOR MAXIMIZE
C     XXMODE=1  FOR MINIMIZE
C     SUBROUTINE MUST BE INITIALIZED BY CALLING IT WITH DELYY=-1.0 AT
C     LEAST ONCE BEFORE IT IS USED FOR A SEARCH
      DELY=DELYY
      A=AA
      B=BB
      XMODE=XXMODE
      IF(DELY) 100,100,101
  100 F(1)=1.0
      F(2)=2.0
      DO 1 I=3,150
      F(I)=F(I-1)+F(I-2)+1.0
      IF(F(I)-1.0E+35) 1,2,2
    2 II=I
      RETURN
    1 CONTINUE
      II=150
      RETURN
  101 YHI=B
      YLO=A
      FNO=(YHI-YLO)/DELY+1.0
      DO 5 N=1,II
      IF(FNO-F(N)) 6,6,5
    5 CONTINUE
```

```
      WRITE(6,7)                                                      DP605770
    7 FORMAT(38H1ERROR, TOO MANY POINTS TO BE SEARCHED)               DP605780
 2002 FORMAT('1',' ERROR HALT')                                       DP605790
 2003 WRITE(6,2002)                                                   DP605800
      STOP                                                            DP605810
    6 IF(N-2) 42,41,40                                                DP605820
   40 Y1=F(N-2)*DELY+YLO                                              DP605830
      CALL FUNCTN(Y1,GY1)                                             DP605840
      Y2=F(N-1)*DELY+YLO                                              DP605850
      CALL FUNCTN(Y2,GY2)                                             DP605860
   16 N=N-1                                                           DP605870
      IF(N-1) 102,102,103                                             DP605880
  103 IF((GY2-GY1)*XMODE) 19,19,20                                    DP605890
   19 YLO=Y1+DELY                                                     DP605900
      Y1=Y2                                                           DP605910
      GY1=GY2                                                         DP605920
      Y2=F(N-1)*DELY+YLO                                              DP605930
      IF(Y2-B) 104,104,105                                            DP605940
  105 GY2=1.0E+35*XMODE                                               DP605950
      GO TO 16                                                        DP605960
  104 CALL FUNCTN(Y2,GY2)                                             DP605970
      GO TO 16                                                        DP605980
   20 YHI=Y2-DELY                                                     DP605990
      Y2=Y1                                                           DP606000
      GY2=GY1                                                         DP606010
      IF(N-2) 34,34,35                                                DP606020
   34 FNM2=0.0                                                        DP606030
      GO TO 36                                                        DP606040
   35 FNM2=F(N-2)                                                     DP606050
   36 Y1=FNM2*DELY+YLO                                                DP606060
      CALL FUNCTN(Y1,GY1)                                             DP606070
      GO TO 16                                                        DP606080
  102 IF((GY2-GY1)*XMODE) 110,111,111                                 DP606090
  110 YYBEST=Y2                                                       DP606100
      BEST=GY2                                                        DP606110
      RETURN                                                          DP606120
  111 YYBEST=Y1                                                       DP606130
      BEST=GY1                                                        DP606140
      RETURN                                                          DP606150
   41 Y1=A                                                            DP606160
      CALL FUNCTN(Y1,GY1)                                             DP606170
      Y2=B                                                            DP606180
      CALL FUNCTN(Y2,GY2)                                             DP606190
      GO TO 102                                                       DP606200
   42 Y1=A                                                            DP606210
      CALL FUNCTN(Y1,GY1)                                             DP606220
      GO TO 111                                                       DP606230
      END                                                             DP606240
```

```
CFUNCTN
      SUBROUTINE FUNCTN(Y,GY)
      DIMENSION FN(10001),FNM1(10001),DNOFXN(1001)
      REAL * 8 TAB(4)
C     REAL * 8 A,B,E,SUMELT,RN,IS,TERM,TEMP,AB,CDF,IDN
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      DN=Y
      NOUTPE=12
C     JUMP MAY BE SET TO 1 IN TRANFM TO INDICATE AN INFEASIBLE XN-1
C     JUMP=0
      CALL TRANFM
      CALL STGRET
C     XMCDE=1 FOR MIN, -1 FOR MAX
C     IF(JUMP.EQ.1) RN=999999*XMODE
      IF (N-1) 146,146,401
  401 K= (YN-YLOW)/DELY+1.001
      IF (K) 212,212,209
  212 WRITE(NOUTPE,210) N,XN,DN
      WRITE(NOUTPE,210) YN
  210 FORMAT(///9H AT STAGEI5,9H WITH XN=E15.8,8H AND DN=E15.8)
 1210 FORMAT(12H XN-1 EQUALSE16.8,21H AND IS OUT OF LIMITS///)
 2002 FORMAT('1',' ERROR HALT')
 2003 WRITE(NOUTPE,2002)
      STOP
  209 IF (K-JTOP) 205,206,212
  206 RNM1=FNM1(JTOP)
      GO TO 207
  205 X=K-1
      X=YLOW+X*DELY
      RNM1=FNM1(K)+(FNM1(K+1)-FNM1(K))*(YN-X)/DELY
  207 GO TO (21,22,23,24),NNFLAG
   21 QN=RN+RNM1
      GO TO 137
   22 QN=RN*RNM1
      GO TO 137
   23 IF (RN-RNM1) 141,141,142
  141 QN=RN
      GO TO 137
  142 QN=RNM1
      GO TO 137
   24 IF (RN-RNM1) 145,145,146
  145 QN=RNM1
      GO TO 137
  146 QN=RN
```

```
  137 GY=ON                                                                    DP606730
      RETURN                                                                   DP606740
      END                                                                      DP606750
CADJUS2                                                                        ADJ00010
      SUBROUTINE ADJUS2                                                        ADJ00020
      DIMENSION FN(10001),FNM1(10001),DNOFXN(1001),A(101),B(101),C(101)      ADJ00030
      DIMENSION D(101),E(101),F(101),G(101),LBOUND(101),T(101),S(101)         ADJ00040
      DIMENSION U(101),V(101)                                                  ADJ00050
      REAL * 8 TAB(4)                                                          ADJ00060
      REAL * 8 INC,LOWER                                                       ADJ00070
      COMMON /CON/TAB                                                          ADJ00080
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER              ADJ00090
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM                     ADJ00100
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3                          ADJ00110
      COMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2                                  ADJ00120
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))                                    ADJ00130
      REWIND KEEP                                                              ADJ00140
      LOUTPE=6                                                                 ADJ00150
      IBALL=11                                                                 ADJ00160
      NOUTPE=12                                                                ADJ00170
      K2=0                                                                     ADJ00180
      FACTOR=2.0                                                               ADJ00190
      C2=10.                                                                   ADJ00200
      READ(KEEP,2)NMAX,MTAPE,SOLVE                                             ADJ00210
      DO 100 L=1,NMAX                                                          ADJ00220
      I=(NMAX+1)-L                                                             ADJ00230
      READ(KEEP,1)N,S(I),V(I),C(I),U(I),T(I)                                   ADJ00240
C     WRITE(IBALL,1)N,S(I),V(I),C(I),U(I),T(I)                                 ADJ00250
  100 CONTINUE                                                                 ADJ00260
      NMAX1=NMAX-1                                                             ADJ00270
      DO 200 I2=1,NMAX1                                                        ADJ00280
C        WRITE(NOUTPE,1)I2,S(I2),V(I2),C(I2),U(I2),T(I2)                       ADJ00290
         CHKLO=V(I2)-U(I2+1)                                                   ADJ00300
         CHKHI=T(I2+1)-V(I2)                                                   ADJ00310
         CHECK=AMIN1(CHKLO,CHKHI)                                              ADJ00320
         IF((CHECK.GE.C(I2+1)).OR.(S(I2+1).EQ.0.))) GO TO 199                  ADJ00330
            K2=K2+1                                                            ADJ00340
            LBOUND(K2)=I2                                                      ADJ00350
  199    CONTINUE                                                              ADJ00360
  200 CONTINUE                                                                 ADJ00370
C     G(1)=V(1)                                                                ADJ00380
C     IF (C(1).LE.100.)FACTOR=10.                                              ADJ00390
C     IF (C(1).LE.50.)FACTOR=100.                                              ADJ00400
      INC=FACTOR*C(2)                                                          ADJ00410
C     TEST=V(1)+INC                                                            ADJ00420
C10   IF(G(1).GE.TEST) GO TO 20                                                ADJ00430
C        G(1)=G(1)+C(1)                                                        ADJ00440
C        GO TO 10                                                              ADJ00450
```

```
C20     F(1)=V(1)-(INT(V(1)/C(1)))*C(1)                                          ADJ00460
 20     F(1)=0.0                                                                  ADJ00470
        G(1)=(INT((V(1)+INC)/C2)+1.)*C2                                           ADJ00480
        W=(INT(V(1)/C2)+1)*C(1)                                                   ADJ00490
C       W=(INT(V(1)/C(1))+1)*C(1)                                                 ADJ00500
C       WRITE(IBALL,21)F(1),V(1),C(1),W                                           ADJ00510
C21     FORMAT(4F10.2)                                                            ADJ00520
C98     WRITE(IBALL,4)F(1),G(1)                                                   ADJ00530
        DO 301 I=2,NMAX1                                                          ADJ00540
C          IF (C(I).LE.100.)FACTOR=10.0                                           ADJ00550
C          IF (C(I).LE.50.)FACTOR=100.0                                           ADJ00560
           INC=FACTOR*C(I+1)                                                      ADJ00570
           F(I)=(INT((V(I)-INC)/C2)+1)*C2                                         ADJ00580
           G(I)=(INT((V(I)+INC)/C2)+1)*C2                                         ADJ00590
C          G(I)=V(I)                                                              ADJ00600
C          G(I)=V(I)                                                              ADJ00610
C          FTEST=V(I)-INC                                                         ADJ00620
C          GTEST=V(I)+INC                                                         ADJ00630
C30        IF (F(I).LE.FTEST) GO TO 60                                            ADJ00640
C45              F(I)=F(I)-C(I)                                                   ADJ00650
C                GO TO 30                                                         ADJ00660
 60        IF ((F(I).GE.F(I-1)).AND.(F(I).GE.0.0)) GO TO 70                       ADJ00670
C50              F(I)=F(I)+C(I)                                                   ADJ00680
 50              F(I)=F(I)+C2                                                     ADJ00690
                 GO TO 60                                                         ADJ00700
C70        IF ((G(I).GE.GTEST).AND.(G(I).GE.G(I-1))) GO TO 80                     ADJ00710
 70        IF (G(I).GE.G(I-1)) GO TO 80                                           ADJ00720
C75              G(I)=G(I)+C(I)                                                   ADJ00730
 75              G(I)=G(I)+C2                                                     ADJ00740
                 GO TO 70                                                         ADJ00750
C          WRITE(NOUTPE,1)I,V(I),XN1,F(I),G(I),FTEST,GTEST                        ADJ00760
 80              IF(G(I).LE.XN1) GO TO 90                                         ADJ00770
C                      G(I)=G(I)-C(I)                                             ADJ00780
                       G(I)=G(I)-C2                                               ADJ00790
                       GO TO 80                                                   ADJ00800
 90              CONTINUE                                                         ADJ00810
 301       CONTINUE                                                               ADJ00820
C       F(NMAX)=XN1                                                               ADJ00830
C       G(NMAX)=XN1                                                               ADJ00840
        F(NMAX)=INT(XN1/C2)*C2                                                    ADJ00850
        G(NMAX)=INT(XN1/C2)*C2                                                    ADJ00860
        IF(K2.NE.0) GO TO 129                                                     ADJ00870
           WRITE(IBALL,131)                                                       ADJ00880
           WRITE(LOUTPE,131)                                                      ADJ00890
           WRITE(NOUTPE,131)                                                      ADJ00900
 131       FORMAT(' NO BOUNDS ARE APPROACHED')                                    ADJ00910
           GO TO 130                                                              ADJ00920
 129    WRITE(IBALL,127)                                                          ADJ00930
```

```
      WRITE(LOUTPE,127)
      WRITE(NOUTPE,127)
127   FORMAT(' THE FOLLOWING STAGES ARE TOO CLOSE TO THEIR BOUNDS')
      WRITE(IBALL,128)(LBOUND(K1),K1=1,K2)
      WRITE(LOUTPE,128) (LBOUND(K1),K1=1,K2)
      WRITE(NOUTPE,128) (LBOUND(K1),K1=1,K2)
128   FORMAT(2X,I3)
130   CONTINUE
      WRITE(NEXT1,2) NMAX,MTAPE,SOLVE
2     FORMAT(2I5,A5)
      K1=1
      F2=0.0
      WRITE(NEXT1,3) K1,K1,F(K1),G(K1),C2,F2,W
      DO 400 K=2,NMAX
      WRITE(NEXT1,3) K,K,F(K),G(K),C2,F(K-1),G(K-1)
400   CONTINUE
3     FORMAT(2I5,3F10.2,'    MIN    SUM   2',2F9.2)
      WRITE(NEXT1,4) F(NMAX),G(NMAX)
4     FORMAT(2F15.2)
1     FORMAT(I5,6F10.2)
6     FORMAT(I5)
7     FORMAT(F10.2)
900   RETURN
      END
CDLIMIT
      SUBROUTINE DLIMIT
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101),F(101),G(101),Q(101),V(101)
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3
      COMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      KEEP=9
      NEXT1=10
      IBALL=11
      NOUTPE=12
      IF(KODE2.NE.0) GO TO 37
      DHIGH=AMIN1((XN-XMIN1L)/C(N)+.0001,160.)
C     IF (DHIGH.GT.Q(N)) Q(N)=DHIGH
      DHIGH=INT(DHIGH)
      DLOW=(XN-XMIN1H)/C(N) +.9999
      DLOW=INT(DLOW)
      DLOW=AMAX1(0.0,DLOW)
C     IF (N.NE.1) GO TO 821
```

```
C820   WRITE(IBALL,819)N,XN,XMIN1L,XMIN1H,C(N),DL)W,DHIGH                DLI00250
C819   FORMAT(1X,I5,6F10.2)                                              DLI00260
C821   CONTINUE                                                          DLI00270
C      DLOW=AMIN1(DLOW,DHIGH)                                            DLI00280
       DEID=1.                                                           DLI00290
       IF((N.GT.1).AND.(C(N).GT.1.0)) GO TO 36                           DLI00300
C      IF(N.EQ.1) DLOW=0.0                                               DLI00310
C      IF (N.EQ.1) DHIGH=10.0                                            DLI00320
C      WRITE(NOUTPE,50)XN,XMIN1L,XMIN1H,C(N),DLOW,DHIGH                  DLI00330
C50    FORMAT(6F10.2)                                                    DLI00340
 36    RETURN                                                            DLI00350
   37  AB=A(N)*B(N)                                                      DLI00360
       DO 555 ID=1,30                                                    DLI00370
       IS=ID-1                                                           DLI00380
C USE THE SECOND ONE FOR DOUBLE PRECISION                                 DLI00390
       TERM=EXP(-AB)                                                     DLI00400
C      TERM=DEXP(-AB)                                                    DLI00410
       TEMP=TERM                                                         DLI00420
       IF(IS.EQ.0) GO TO 11                                              DLI00430
       DO 10 I=1,IS                                                      DLI00440
       TEMP=TEMP*AB/I                                                    DLI00450
       IF(TERM.GE..99999) GO TO 11                                       DLI00460
C      IF(TEMP.LE..00001) GO TO 11                                       DLI00470
   10  TERM=TERM +TEMP                                                   DLI00480
   11  CDF=TERM                                                          DLI00490
   40  TWUS=(1.-CDF)*(AB*AB-2.*AB*IS+IS*(IS+1))/(2.*A(N))                DLI00500
      X+TEMP*B(N)*(AB-IS)/2.                                             DLI00510
       IF(KODE2.EQ.2.AND.E(N)*TWUS/SUMELT.LE.XN)3) TO 556                DLI00520
C  CHANGED FOLLOWING TO 555 FROM 556                                     DLI00530
       IF(KODE2.EQ.2)GO TO 555                                           DLI00540
       IF(KODE2.EQ.3)GO TO 554                                           DLI00550
       SSS=(1.-TEMP)+(IS-AB)*(1.-CDF)/AB                                 DLI00560
       RR=E(N)*AB*SSS/SUMELT                                             DLI00570
       Y=XN-RR                                                           DLI00580
       DLCW=0.                                                           DLI00590
C THE NUMBER IN THE NEXT LINE HAS A BIG EFFECT. [T SHOULD NOT.            DLI00600
       IF(N.EQ.1) DLOW=1000.                                             DLI00610
       IF(N.EQ.1.AND.Y.GT.0..AND.ID.LT.30)GO TO 555                      DLI00620
       IF(N.EQ.1.AND.Y.LE.0.)DLOW=IS                                     DLI00630
  136  DHIGH=30.+DLOW                                                    DLI00640
       IF(DLOW.EQ.1000.)DHIGH=1000.                                      DLI00650
       DEID=1.                                                           DLI00660
       RETURN                                                            DLI00670
  554  AMSRT=TWUS/AB                                                     DLI00680
       AMTBF=1./A(N)                                                     DLI00690
       AMTTR=D(N)                                                        DLI00700
       AV=AMTBF/(AMTBF+AMTTR+AMSRT)                                      DLI00710
       IF(AV.GE.XN) GC TO 556                                            DLI00720
```

```
  555 CONTINUE
      IS=30.
  556 DLOW=IS
      DELD=1.
      DHIGH=50.
      RETURN
      END
      BLOCK DATA
      COMMON /CON/TAB
      REAL * 8 TAB(4) /'   SUM','  MULT','MAXMIN','MINMAX'/
      END
CENTER2
      SUBROUTINE ENTER2
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101),F(101),G(101),S(101),V(101)
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF
      REAL * 8 TAB(4)
      REAL INC
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3
      COMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      KEEP=9
      NEXT1=10
      IBALL=11
      NOUTPE=12
      INDEC=14
      IFILE=15
      FACTOR=2.0
      NMAX1=NMAX-1
      TOTAL=0.0
      READ (INDEC,2) NMAX,MTAPE,SOLVE,ITOTAL
C     WRITE(IBALL,14)NMAX
   14 FORMAT(I5)
      DO 100 I=1,NMAX
      READ (INDEC,11) S(I)
   11 FORMAT(F10.2)
      POINT=S(I)*C(I)
      TOTAL=TOTAL+POINT
C     WRITE(IBALL,11)TOTAL
      V(I)=TOTAL
  100 CONTINUE
      F(NMAX)=V(NMAX)
      G(NMAX)=V(NMAX)
      IF(ITOTAL.EQ.0) GO TO 9
      READ (INDEC,4) XN1,XN2
```

```
      F(NMAX)=XN1
      G(NMAX)=XN2
    9 G(1)=V(1)
C     IF (C(1).LE.100.) FACTOR=10.
C     IF (C(1).LE.50.) FACTOR=100.
      INC=FACTOR*C(2)
      TEST=V(1)+INC
C     WRITE(IBALL,12)S(1),C(1),TOTAL,V(1),TEST,INC
   12 FORMAT(6F10.2)
   10 IF(G(1).GE.TEST) GO TO 20
         G(1)=G(1)+C(1)
         GO TO 10
   20 F(1)=V(1)-(INT(V(1)/C(1)))*C(1)
      W=(INT(V(1)/C(1))+1)*C(1)
C98   WRITE(IBALL,4)F(1),G(1)
      DO 301 J=2,NMAX1
C        IF (C(J).LE.100.) FACTOR=10.0
C        IF (C(J).LE.50.) FACTOR=100.0
         INC=FACTOR*C(J+1)
         F(J)=V(J)
         G(J)=V(J)
         FTEST=V(J)-INC
         GTEST=V(J)+INC
   30    IF (F(J).LE.FTEST) GO TO 60
   45          F(J)=F(J)-C(J)
               GO TO 30
   60    IF ((F(J).GE.F(J-1)).AND.(F(J).GE.0.0)) GO TO 70
   50          F(J)=F(J)+C(J)
               GO TO 60
   70    IF ((G(J).GE.GTEST).AND.(G(J).GE.G(J-1))) GO TO 80
   75          G(J)=G(J)+C(J)
               GO TO 70
   80          IF (G(J).LE.XN1) GO TO 90
                     G(J)=G(J)-C(J)
                     GC TO 80
   90          CONTINUE
C     WRITE(IBALL,12)S(J),C(J),TOTAL,V(J),FTEST,GTEST,INC
   13 FORMAT(7F8.2)
  301 CCNTINUE
      IF(ITOTAL.EQ.0) GO TO 99
      G(NMAX1)=XN2
   99 CONTINUE
      WRITE(IFILE,2) NMAX,MTAPE,SOLVE,ITOTAL
    2 FORMAT(2I5,A5,I5)
      K1=1
      F2=0.0
      WRITE(IFILE,3) K1,K1,F(K1),G(K1),C(K1),F2,W
      DO 400 K=2,NMAX
```

```
      WRITE(IFILE,3) K,K,F(K),G(K),C(K),F(K-1),G(K-1)                   ENT00860
  400 CONTINUE                                                          ENT00870
    3 FORMAT(2I5,3F10.2,'    MIN    SUM   2',2F9.2)                     ENT00880
      WRITE(IFILE,4) F(NMAX),G(NMAX)                                    ENT00890
    4 FORMAT(2F15.2)                                                    ENT00900
    1 FORMAT(I5,5F10.2)                                                 ENT00910
    6 FORMAT(I5)                                                        ENT00920
    7 FORMAT(F10.2)                                                     ENT00930
  900 RETURN                                                            ENT00940
      END                                                               ENT00950
CSTGRET                                                                 STG00010
      SUBROUTINE STGRET                                                 STG00020
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)STG00030
      DIMENSION D(101),E(101),F(101),G(101),V(101)                      STG00040
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF                         STG00050
      REAL * 8 TAB(4)                                                   STG00060
      COMMON /CON/TAB                                                   STG00070
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER       STG00080
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM              STG00090
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3                   STG00100
      COMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2                           STG00110
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))                             STG00120
    1 CONTINUE                                                          STG00130
      NEXT1=10                                                          STG00140
      IBALL=11                                                          STG00150
      NOUTPE=12                                                         STG00160
      IF(KODE1.EQ.0) GO TO 77                                           STG00170
      IDN=INT(DN)                                                       STG00180
      AB=A(N)*B(N)                                                      STG00190
      IS=IDN                                                            STG00200
      TERM=0.                                                           STG00210
      TEMP=0.                                                           STG00220
      IF(IS.LT.0) GO TO 11                                              STG00230
      IF(AB.LE.170.0) GO TO 12                                          STG00240
      TERM=0.0                                                          STG00250
      GO TO 13                                                          STG00260
C USE THE SECOND ONE FOR DOUBLE PRECISION                               STG00270
   12 TERM=EXP(-AB)                                                     STG00280
C12   TERM=DEXP(-AB)                                                    STG00290
   13 TEMP=TERM                                                         STG00300
      IF(IS.EQ.0) GO TO 11                                              STG00310
      DO 10 I=1,IS                                                      STG00320
      TEMP=TEMP*AB/I                                                    STG00330
C     CALL OVERFL (JOY)                                                 STG00340
C     IF (JOY.NE.2) WRITE(6,999) TEMP,TERM,A(N),B(N),AB,I,IS            STG00350
C999  FORMAT(1X,5E13.5,2I5)                                             STG00360
      IF(TERM.GE..99999) GO TO 11                                       STG00370
C     IF(TEMP.LE..00001) GO TO 11                                       STG00380
```

```
   10 TERM=TERM  +TEMP                                                   STG00390
   11 CDF=TERM                                                           STG00400
   20 CONTINUE                                                           STG00410
      IF(KODE1.NE.1) GO TO 40                                            STG00420
C FOLLOWING CHANGED 120183                                                STG00430
C     SMA=(AB*(1.-TEMP)+(IDN-AB)*(1.-CDF))/AB                            STG00440
      SMA=(1.-TEMP)+(IDN-AB)*(1.-CDF)/AB                                 STG00450
      RN=E(N)*AB*SMA/SUMELT                                              STG00460
      RETURN                                                             STG00470
   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*IDN+IDN*(IDN+1))/(2.*A(N))              STG00480
     X+TEMP*B(N)*(AB-IDN)/2.                                             STG00490
C     WRITE(NEXT1,101)KODE1                                              STG00500
C101  FORMAT(I3)                                                         STG00510
      IF(KODE1.EQ.3)GO TO 50                                             STG00520
 47   IF (XN.GE.XMIN1L) GO TO 48                                         STG00530
C     WRITE(NEXT1,100)XN,XMIN1L                                          STG00540
C100  FORMAT(2F10.2)                                                     STG00550
         RN=1.0*MODES*10.0E5                                             STG00560
         GO TO 49                                                        STG00570
 48   RN=E(N)*TWUS/SUMELT                                                STG00580
 148  RN1=E(N)*TWUS                                                      STG00590
C     IF((N.LT.3).AND.(C(N).LT.15.0)) GO TO 103                          STG00600
C     WRITE(IBALL,102)XN,DN,RN,RN1                                       STG00610
C102  FORMAT(6X,2F10.2,2F15.9)                                           STG00620
C103  CONTINUE                                                           STG00630
 49   RETURN                                                             STG00640
   50 CONTINUE                                                           STG00650
      AMSRT=TWUS/AB                                                      STG00660
      AMTBF=1./A(N)                                                      STG00670
      AMTTR=D(N)                                                         STG00680
      RN=(AMTBF)/(AMTBF+AMTTR+AMSRT)                                     STG00690
      RETURN                                                             STG00700
 77   RN=C(N)*DN                                                         STG00710
C     IF(KODE2.EQ.1.AND.N.EQ.1.AND.YN.LT.XN)RN=9)9999                    STG00720
 80   RETURN                                                             STG00730
      END                                                                STG00740
CSTORE                                                                   STO00010
      SUBROUTINE STORE                                                   STO00020
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101) STO00030
      DIMENSION D(101),E(101),F(101),G(101),V(101)                       STO00040
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF                          STO00050
      REAL * 8 TAB(4)                                                    STO00060
      COMMON /CON/TAB                                                    STO00070
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,)HIGH,DELD,NUMBER        STO00080
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM               STO00090
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3                    STO00100
      CCMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2                            STO00110
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))                              STO00120
```

```
      INDATA=8
      NOUTPE=12
      LOUTPE=6
      XMIN1L=0.0
      XMIN1H=0.0
      READ (INDATA,808) KODE1,KODE2,KODE3
      WRITE(NOUTPE,808) KODE1,KODE2,KODE3
  808 FORMAT (3I3)
      IF(KODE1.EQ.0) WRITE(NOUTPE,110)
      IF(KODE1.EQ.1) WRITE(NOUTPE,101)
      IF(KODE1.EQ.2) WRITE(NOUTPE,202)
      IF(KODE1.EQ.2) WRITE(LOUTPE,202)
      IF(KODE1.EQ.3) WRITE(NOUTPE,303)
  110 FORMAT (30H THE OBJECTIVE IS COST          )
  101 FORMAT (30H THE OBJECTIVE IS SMA           )
  202 FORMAT (30H THE OBJECTIVE IS MSRT          )
  303 FORMAT (30H THE OBJECTIVE IS AVAILABILITY)
      IF(KODE2.EQ.0) WRITE(NOUTPE,900)
      IF(KODE2.EQ.0) WRITE(LOUTPE,900)
      IF(KODE2.EQ.1) WRITE(NOUTPE,901)
      IF(KODE2.EQ.2) WRITE(NOUTPE,902)
      IF(KODE2.EQ.3) WRITE(NOUTPE,903)
  900 FORMAT (30H CONSTRAINT ON COST           )
  901 FORMAT (30H CONSTRAINT ON SMA            )
  902 FORMAT (30H CONSTRAINT ON MSRT           )
  903 FORMAT (30H CONSTRAINT ON AVAILABILITY   )
C     WRITE(NOUTPE,1)NMAX
C1    FORMAT(I5)
      DO 333 K=1,NMAX
      READ (INDATA,999) A(K),C(K),E(K),B(K),D(K)
  999 FORMAT (5F10.4)
C     WRITE(NOUTPE,999) A(K),B(K),C(K),D(K),E(K)
  333 CONTINUE
      SUMELT=0.
      DO 30 I=1,NMAX
   30 SUMELT=SUMELT+E(I)*A(I)*B(I)
      WRITE(NOUTPE,765)SUMELT
  765 FORMAT(F10.3)
      RETURN
      END
CTRANFM
      SUBROUTINE TRANFM
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101),F(101),G(101),V(101)
C     REAL * 8 A,B,E,SUMELT,RN,TERM,TEMP,AB,CDF
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,XLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
```

```
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM              TRA00090
      COMMON A,B,C,D,E,F,G,V,SUMELT,KODE1,KODE2,KODE3                   TRA00100
      CCMMON XMIN1L,XMIN1H,KEEP,NEXT1,XN1,XN2                           TRA00110
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))                             TRA00120
      IF(KODE2.NE.0) GC TO 12                                           TRA00130
      IF (XN.GE.XMIN1L) GO TO 30                                        TRA00140
          YN=XMIN1L                                                     TRA00150
          GO TO 31                                                      TRA00160
 30   YN=XN-C(N) *DN                                                    TRA00170
 31   RETURN                                                            TRA00180
   12 IDN=INT(DN)                                                       TRA00190
      AB=A(N)*B(N)                                                      TRA00200
      IS=IDN                                                            TRA00210
      TERM=0.                                                           TRA00220
      TEMP=0.                                                           TRA00230
      IF(IS.LT.0) GO TO 11                                              TRA00240
C USE THE SECOND ONE FOR DOUBLE PRECISION                               TRA00250
      TERM=EXP(-AB)                                                     TRA00260
C     TERM=DEXP(-AB)                                                    TRA00270
      TEMP=TERM                                                         TRA00280
      IF(IS.EQ.0) GO TO 11                                              TRA00290
      DO 10 I=1,IS                                                      TRA00300
      TEMP=TEMP*AB/I                                                    TRA00310
      IF(TERM.GE..99999) GO TO 11                                       TRA00320
C     IF(TEMP.LE..00001) GO TO 11                                       TRA00330
   10 TERM=TERM +TEMP                                                   TRA00340
   11 CDF=TERM                                                          TRA00350
   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*IDN+IDN*(IDN+1) )/(2.*A(N))            TRA00360
     X+TEMP*B(N)*(AB-IDN)/2.                                            TRA00370
      IF(KODE2.EQ.2) GO TO 50                                           TRA00380
      IF(KODE2.EQ.3) GO TO 51                                           TRA00390
      SMA=(1.-TEMP)+(IDN-AB)*(1.-CDF)/AB                                TRA00400
      YN=XN-E(N)*AB*SMA/SUMELT                                          TRA00410
                                                                        TRA00420
      IF(YN.LT.0) YN=0.0                                                TRA00430
                                                                        TRA00440
C     IF(N.NE.1.AND.YN.LT.0.OR.N.NE.1.AND.YN.GT.1.) JUMP=1              TRA00450
C     IF(N.EQ.1.AND.YN.GT.0)JUMP=1                                      TRA00460
C     IF(JUMP.EQ.1) YN=0.                                               TRA00470
      RETURN                                                            TRA00480
   51 AMSRT=TWUS/AB                                                     TRA00490
      AMTBF=1./A(N)                                                     TRA00500
      AMTTR=D(N)                                                        TRA00510
      AV=(AMTBF)/(AMTBF+AMTTR+AMSRT)                                    TRA00520
      YN=XN/AV                                                          TRA00530
C     AV IS AVAIL                                                       TRA00540
      RETURN                                                            TRA00550
   50 YN=XN-E(N)*TWUS/SUMELT                                            TRA00560
```

```
      IF(YN.LE.0.)YN=0.                                              TRA00570
      RETURN                                                         TRA00580
      END                                                            TRA00590
```

LIST OF REFERENCES

1. Naval Postgraduate School Report NPS55-83-026, Wholesale Provisioning Models: Model Development, by F. Russell Richards and Alan W. McMasters, September 1983.

2. Department of Defense Instruction 4140.42, "Determination of Initial Requirements for Secondary Item Spares and Repair Parts", 7 August 1974.

3. Naval Postgraduate School Report NPS55-83-028, Wholesale Provisioning Models: Model Optimization, by Gilbert T. Howard, October 1983.

4. Fox, Bennett, "Discrete Optimization Via Marginal Analysis", Management Science, Vol 13, No. 3, pp. 210-215, November 1966.

5. Nemhauser, George, L., Introduction to Dynamic Programming, pp. 22-23, John Wiley and Sons, Inc., New York, 1966.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Graduate School Monterey, California 93943	2
3.	Gilbert T. Howard Naval Postgraduate School Code 55HK Monterey, California 93943	2
4.	F. Russell Richards Naval Postgraduate School Code 55RH Monterey, California 93943	2
5.	LCDR. William A. Goulding 409 Camino Elevado Bonita, California 92002	3

END

FILMED

5-85

DTIC